

ter Meer, Steinmeister & Partner GbR  
Einspruch gegen EP 1 830 241 B1  
Patentinhaberin: Rambus Inc.  
Einsprechende:  
Hynix Semiconductor Deutschland GmbH  
Dokument D10

⑮ BUNDESREPUBLIK  
DEUTSCHLAND



DEUTSCHES  
PATENTAMT

⑫ Übersetzung der  
europäischen Patentschrift

② EP 0 346 420 B1

⑩ DE 38 50 770 T 2

G 11 C 8/00

DE 38 50 770 T 2

②	Deutsches Aktenzeichen:	38 50 770.6
⑧	PCT-Aktenzeichen:	PCT/FR88/00608
⑨	Europäisches Aktenzeichen:	89 900 274.5
⑦	PCT-Veröffentlichungs-Nr.:	WO 89/08013
⑥	PCT-Anmeldetag:	9. 12. 88
⑤	Veröffentlichungstag der PCT-Anmeldung:	29. 6. 89
④	Erstveröffentlichung durch das EPA:	20. 12. 89
③	Veröffentlichungstag der Patenterteilung beim EPA:	20. 7. 94
①	Veröffentlichungstag im Patentblatt:	19. 1. 95

⑬ Unionspriorität: ⑬ ⑭ ⑮  
14.12.87 FR 8718103

⑰ Patentinhaber:  
Busless Computers, S.a.r.l., Toulouse, FR

⑱ Vertreter:  
Weitzel, W., Dipl.-Ing. Dr.-Ing., Pat.-Anw., 89522  
Heidenheim

⑲ Benannte Vertragsstaaten:  
DE, FR, GB, IT, NL

⑲ Erfinder:  
LITAIZE, Daniel, F-31650 Saint-Orens-de-Gameville,  
FR; SALINIER, Jean-Claude, F-31520  
Ramonville-Saint-Agne, FR; MZOUGH, Abdelaziz,  
F-31100 Toulouse, FR; ELKHLIFI, Fatima-Zahra,  
F-31400 Toulouse, FR; LALAM, Mustapha, F-31000  
Toulouse, FR; SAINRAT, Pascal, F-31000 Toulouse,  
FR

⑲ VERFAHREN ZUM INFORMATIONSAUSTAUSCH IN EINEM MEHRPROZESSORSYSTEM.

Anmerkung: Innerhalb von neun Monaten nach der Bekanntmachung des Hinweises auf die Erteilung des europäischen Patents kann jedermann beim Europäischen Patentamt gegen das erteilte europäische Patent Einspruch einlegen. Der Einspruch ist schriftlich einzureichen und zu begründen. Er gilt erst als eingelegt, wenn die Einspruchsgebühr entrichtet worden ist (Art. 99 (1) Europäisches Patentübereinkommen).

Die Übersetzung ist gemäß Artikel II § 3 Abs. 1 IntPatÜG 1991 vom Patentinhaber eingereicht worden. Sie wurde vom Deutschen Patentamt inhaltlich nicht geprüft.

DE 38 50 770 T 2

EP 0 346 420

Anwaltsakte : P 13038 1

## Verfahren zum Informationsaustausch in einem Mehrprozessorsystem

Die Erfindung betrifft ein Mehrprozessorsystem  
5 mit einem Hauptspeicher, Verarbeitungs-Prozessoren und den  
Verarbeitungs-Prozessoren zugeordnete Cache-Speicher. Sie  
erstreckt sich weiterhin auf ein Verfahren zum  
Informationsaustausch zwischen einem Hauptspeicher und  
Verarbeitungs-Prozessoren über den jedem dieser  
10 Prozessoren zugeordneten Cache-Speicher. Ihr Ziel ist auch  
ein neues integriertes Schaltkreisbauteil, mit dem das  
Mehrprozessorsystem ausgerüstet werden kann.

Es ist bekannt, daß in den gebräuchlichsten  
Mehrprozessorsystemen alle Informationen (Daten, Adressen)  
15 zwischen dem Hauptspeicher und den verschiedenen  
Verarbeitungs-Prozessoren durch einen gemeinsamen  
parallelen Kommunikationsbus übertragen werden, der einen  
Engpaß bildet: seine Übertragungsrate ist in der Tat nicht  
ausreichend, um alle Prozessoren bei voller Leistung von  
20 einem gemeinsamen Hauptspeicher aus zu versorgen.

Um die Übertragungsrate der Informationen zu  
erhöhen, besteht eine erste Lösung darin, jedem  
Verarbeitungs-Prozessor einen Cache-Speicher zuzuordnen,  
der durch die Lage der Information eine Verringerung der  
25 Anforderungen an den Hauptspeicher ermöglicht. Ist das  
Volumen der zwischen den Prozessoren geteilten Daten  
jedoch beträchtlich, dann erzeugt die Aufrechterhaltung  
der Kohärenz der Daten zwischen den Speichern einen  
zusätzlichen Informationsverkehr auf dem  
30 Kommunikationsbus, was sich einer bedeutenden Verringerung  
der globalen Datenrate auf diesem Bus entgegenstellt, und  
diese Lösung somit einen großen Teil ihres Interesses  
einbüßt.

Eine andere Lösung besteht darin, den  
35 Kommunikationsbus in Form eines vermaschten Netzes,  
"crossbar" genannt, auszuführen, was eine direkte  
Kommunikation zwischen jedem Verarbeitungs-Prozessor und  
jeder Unterbaugruppe des Hauptspeichers (Speicherbank)

ermöglicht. Jedoch ist diese Lösung in der Ausführung aufgrund der sehr hohen Anzahl von Querverbindungen sehr schwerfällig und sehr kostspielig, und sie wird vollkommen unrealistisch, wenn die Anzahl der Verarbeitungs-  
 5 Prozessoren zehn übersteigt. Außerdem impliziert eine solche Lösung im Falle vielfacher Anforderungen von mehreren Prozessoren an eine selbe Speicherbank Zugriffskonflikte, die eine Quelle von Austauschverzögerungen darstellen.

10 Eine andere, aufgrund der Einfachheit ihrer Architektur gängigere Lösung besteht darin, jedem Verarbeitungs-Prozessor einen Lokalspeicher zuzuordnen, um die ihn betreffenden Daten zu speichern und um die geteilten Daten in dem gemeinsamen Hauptspeicher zu  
 15 speichern. Jedoch besteht der große Nachteil dieser Architektur in ihrer mangelnden Transparenz, das heißt in der Notwendigkeit für den Programmierer, die Zuordnungen der Daten in die verschiedenen Speicher im Detail zu organisieren, und zwar in dem Maße, daß die Anwendung  
 20 dieser Lösung ausgesprochen einengend ist. Außerdem kann sie, im Fall eines hohen Volumens geteilter Daten, wie im vorhergehenden Fall zu einer Sättigung des Busses für den Zugriff auf den Hauptspeicher führen.

Im übrigen wurde von der Berkeley-Universität  
 25 eine sogenannte "aquarius Architektur" vorgeschlagen, die darin besteht, die oben genannte Crossbar-Lösung dadurch zu verbessern, daß einerseits, was die nicht geteilten Daten betrifft, an das Crossbar-Netz angeschlossene Cache-Speicher und andererseits, was die geteilten Daten  
 30 betrifft, andere, an einen gemeinsamen Synchronisationsbus angeschlossene Cache-Speicher mit dem Crossbar-Netz kombiniert werden. Diese Lösung bringt eine schnellere Austausch-Geschwindigkeit, bleibt aber sehr schwerfällig und sehr kostspielig in der Ausführung.

35 Die vorliegende Erfindung möchte eine neue Lösung liefern, die es ermöglicht, die Informationsaustauschraten beträchtlich zu erhöhen, und zwar unter Wahrung einer für den Benutzer transparenten

Architektur, die sehr viel einfacher ist, als die Crossbar-Architektur.

Ein Ziel der Erfindung ist somit, eine merkliche Erhöhung der Anzahl der Verarbeitungs-  
 5 Prozessoren des Systems zu ermöglichen, wobei ein hoher Wirkungsgrad für jeden Prozessor ausgenutzt wird.

Ein anderes Ziel ist es, eine Struktur eines integrierten Schaltkreisbauteils zu liefern, die eine sehr einfache Durchführung der Architektur dieses neuen  
 10 Mehrprozessorsystems ermöglicht.

Zu diesem Zweck enthält das von der Erfindung angestrebte Mehrprozessorsystem einen Hauptspeicher (RAM), der in Informationsblöcken (bi) angeordnet ist, Verarbeitungs-Prozessoren ( $CPU_1 \dots CPU_j \dots CPU_n$ ), einen  
 15 Cache-Speicher ( $MC_j$ ), der mit jedem Verarbeitungs-Prozessor ( $CPU_j$ ) verbunden und in Informationsblöcken (bi) angeordnet ist, die genauso groß sind, wie die des Hauptspeichers, ein Verzeichnis ( $RG_j$ ) und dessen Verwaltungsprozessor ( $PG_j$ ), die beide jedem Cache-Speicher  
 20 ( $MC_j$ ) zugeordnet sind, Mittel für die Übertragung der Blockadressen zwischen Prozessoren ( $CPU_j$ ) und Hauptspeicher (RAM); erfindungsgemäß ist das besagte Mehrprozessorsystem ausgestattet mit:

mehreren Schieberegistern, sogenannten  
 25 Speicher-Schieberegistern ( $RDM_1 \dots RDM_j \dots RDM_n$ ), wobei jedes dieser Register ( $RDM_j$ ) so an den Hauptspeicher (RAM) angeschlossen ist, daß innerhalb eines Zyklus dieses Speichers ein paralleler Lese- oder Schreibtransfer eines Informationsblocks (bi) zwischen dem besagten Register und  
 30 dem besagten Hauptspeicher möglich ist,

Schieberegistern, sogenannten Prozessor-Schieberegistern ( $RDP_1 \dots RDP_j \dots RDP_n$ ), wobei jedes Prozessor-Schieberegister ( $RDP_j$ ) so mit dem Cache-Speicher ( $MC_j$ ) eines Prozessors ( $CPU_j$ ) verbunden ist, daß ein  
 35 paralleler Lese- oder Schreibtransfer eines Informationsblocks (bi) zwischen dem besagten Schieberegister ( $RDP_j$ ) und dem besagten Cache-Speicher ( $MC_j$ ) möglich ist,

. mehreren Serienverbindungen ( $LS_1 \dots LS_j$ ,  
 $\dots LS_n$ ), wobei jede ein Speicher-Schieberegister ( $RDM_j$ )  
 mit einem Prozessor-Schieberegister ( $RDP_j$ ) verbindet und  
 geeignet ist, den Transfer von Informationsblöcken ( $bi$ )  
 5 zwischen den beiden betreffenden Registern ( $RDM_j$ ,  $RDP_j$ ) zu  
 ermöglichen.

So erfolgt in dem erfindungsgemäßen  
 Mehrprozessorsystem der Austausch zwischen Cache-Speichern  
 und zugeordneten Prozessoren wie in den herkömmlichen, mit  
 10 Cache-Speichern ausgestatteten Systemen. Im Gegensatz  
 dazu, erfolgt der Austausch zwischen Hauptspeicher und  
 Cache-Speichern gänzlich neuartig.

Jeder Transfer eines Informationsblocks ( $bi$ )  
 vom Hauptspeicher (RAM) zum Cache-Speicher ( $MC_j$ ) eines  
 15 gegebenen Prozessors ( $CPU_j$ ) besteht darin:

. innerhalb eines Zyklus des Hauptspeichers  
 den Block ( $bi$ ) des besagten Hauptspeichers (RAM) zum  
 Speicher-Schieberegister ( $RDM_j$ ) (von der Größe eines  
 Blocks) zu übertragen, das direkt an den Hauptspeicher  
 20 angeschlossen ist und das dem betreffenden Prozessor  
 ( $CPU_j$ ) entspricht,

. auf der entsprechenden Serienverbindung  
 ( $LS_j$ ) den Inhalt dieses Speicher-Schieberegisters ( $RDM_j$ )  
 zum Prozessor-Schieberegister ( $RDP_j$ ) (von der gleichen  
 25 Kapazität) zu übertragen, das dem Cache-Speicher ( $MC_j$ ) des  
 betreffenden Prozessors ( $CPU_j$ ) zugeordnet ist,

. den Inhalt des besagten Prozessor-  
 Schieberegisters ( $RDP_j$ ) zu dem besagten Cache-Speicher  
 ( $MC_j$ ) zu übertragen.

30 In der entgegengesetzten Richtung besteht  
 jeder Transfer eines Informationsblocks ( $bi$ ) vom Cache-  
 Speicher ( $MC_j$ ) eines gegebenen Prozessors ( $CPU_j$ ) zum  
 Hauptspeicher (RAM) darin:

. den Block ( $bi$ ) von dem besagten betreffenden  
 35 Cache-Speicher ( $MC_j$ ) zum Prozessor-Schieberegister ( $RDP_j$ )  
 zu übertragen, das dem besagten Cache-Speicher ( $MC_j$ )  
 zugeordnet ist,

. den Inhalt des Prozessor-Schieberegisters ( $RDP_j$ ) auf der entsprechenden Serienverbindung ( $LS_j$ ) zum Speicher-Schieberegister ( $RDM_j$ ) zu übertragen, das dem betreffenden Prozessor zugeordnet ist (von allen  
5 Schieberegistern ( $RDM_1 \dots RDM_j \dots RDM_n$ ), die an den Hauptspeicher (RAM) angeschlossen sind),

. den Inhalt des Speicher-Schieberegisters ( $RDM_j$ ) innerhalb eines Zyklus des Hauptspeichers zu dem besagten Hauptspeicher (RAM) zu übertragen.

10        Unter diesen Bedingungen erfolgt der Transfer aller Informationsblöcke ( $bi$ ) nicht mehr durch einen parallelen Bus, wie es bei den bekannten Systemen der Fall ist, sondern über Serienverbindungen mit einer hohen Übertragungsgeschwindigkeit. Diese Serienverbindungen  
15 ermöglichen Transferdauern eines Blocks ( $bi$ ), die vergleichbar sind mit den Transferdauern in den bekannten Systemen mit parallelem Bus oder sogar geringer, als diese. Das unten gegebene vergleichende Beispiel mit für die aktuelle Technologie gängigen Parameterwerten zeigt  
20 klar und deutlich diese paradox erscheinende Tatsache.

Es wird davon ausgegangen, daß jeder Informationsblock ( $bi$ ) 64 Byte groß ist.

In dem erfindungsgemäßen System teilt sich die Transferdauer zwischen dem Hauptspeicher und einem Cache-Speicher auf in:  
25

- eine Hauptspeicher (RAM)/Speicher-Schieberegister ( $RDM_j$ )-Transferdauer: 100 Nanosekunden (Leistung eines gängigen Hauptspeichers mit Zufallsreihenfolgezugriff),
- 30        - eine Serientransferdauer auf der entsprechenden Serienverbindung:  $64 \times 8 \times 1/500.10^6$ , das sind 1 024 Nanosekunden, wobei eine Transferfrequenz von 500 Megahertz angenommen wird (was bei den aktuellen Technologien, die Frequenzen bis zu 3 000 Megahertz  
35 zulassen, nicht außergewöhnlich ist),
- eine Prozessor-Schieberegister ( $RDP_j$ )/Cache-Speicher ( $MC_j$ )-Transferdauer: 50 Nanosekunden (sehr gängiger Cache-Speicher).

Die gesamte Transferdauer eines Blocks beträgt somit circa 1 200 Nanosekunden (wobei zweitrangige Verkettungs-Zeiten in Betracht gezogen werden).

Bei den bekannten Systemen mit Cache-Speichern, bei denen der Informationsaustausch direkt parallel durch 4-Byte-Wörter erfolgt (die gängigsten Systeme, die zu gewöhnlichen Bussen mit 32 Datendrähten führen), ist die Transferdauer eines Blocks gleich der Transferdauer der 16 4-Byte-Wörter, aus denen dieser Block besteht, das heißt:  $16 \times 100 = 1\,600$  Nanosekunden.

Man sieht also, daß diese Zeiten, bei durchschnittlichen Annahmen für beide Lösungen, vergleichbar sind. Beim Vergleich der Architektur des erfindungsgemäßen Systems mit der mit einem parallelen gemeinsamen Bus mit Cache-Speichern (die erste zuvor genannte Lösung) stellt man jedoch fest, daß:

. bei der herkömmlichen Lösung (gemeinsamer paralleler Bus) der Hauptspeicher und der gemeinsame Bus während des Transfers zu 100 % belegt sind, da zwischen den beiden während der gesamten Transferdauer Informationen ausgetauscht werden,

. bei dem erfindungsgemäßen System die Serienverbindung während des Transfers zu 100 % belegt ist, der Hauptspeicher jedoch zu weniger als 10 % der Transferdauer belegt ist (Lese-Dauer des Speichers und Lade-Dauer des Speicher-Schieberegisters ( $RDM_1$ )), so daß der Hauptspeicher 10 mal mehr Prozessoren bedienen kann, als im vorhergehenden Fall (wobei die Belegung der Serienverbindung ohne Bedeutung ist, da sie privat und dem Prozessor zugeordnet ist).

Es muß im übrigen hervorgehoben werden, daß in dem erfindungsgemäßen System jede Serienverbindung, die jeden einzelnen Prozessor mit dem Hauptspeicher verbindet, eine einfache Verbindung ist (mit einem oder zwei Datendrähten), sodaß das so gebildete Seriennetz von der Komplexität her zum Beispiel nicht mit einem Crossbar-Netz vergleichbar ist, bei dem jede Verbindung eine Parallelverbindung mit einer Vielzahl von Drähten (32

Datendrähte bei dem vorgenannten vergleichenden Beispiel) mit allen notwendigen Schaltern ist.

Außerdem besitzt das erfindungsgemäße System, wie man später auf den vergleichenden Kurven sehen wird, deutlich verbesserte Leistungen im Vergleich zu den traditionellen Systemen mit einem gemeinsamen Bus und ermöglicht in der Praxis den Einsatz einer sehr viel höheren Anzahl von Verarbeitungs-Prozessoren (von mehreren zehn bis zu circa hundert Prozessoren); diese Leistungen sind kompatibel mit denen eines Crossbar-Systems, doch besitzt das erfindungsgemäße System eine sehr viel einfachere Architektur.

Bei dem erfindungsgemäßen System kann jede Serienverbindung praktisch entweder mittels zweier unidirektionaler Seriendrähte für einen bit-für-bit-Transfer durchgeführt werden oder mittels eines einzigen bidirektionalen Drahts.

Im ersten Fall wird jedes Speicher-Schieberegister ( $RDM_j$ ) und jedes Prozessor-Schieberegister ( $RDP_j$ ) in zwei Register aufgeteilt, das eine ist auf den Transfer in die eine Richtung, das andere auf den Transfer in die andere Richtung spezialisiert. Die beiden unidirektionalen Seriendrähte werden dann so an das aufgeteilte Speicher-Schieberegister ( $RDM_j$ ) und das entsprechende aufgeteilte Prozessor-Schieberegister ( $RDP_j$ ) angeschlossen, daß das eine den Transfer in die eine Richtung und das andere den Transfer in die andere Richtung ermöglicht.

Diese Art der Durchführung mit zwei unidirektionalen Drähten bietet den Vorteil, daß keine Transferverwaltung auf den Drähten notwendig ist, jedoch besteht der Nachteil, daß die notwendigen Betriebsmittel (Draht, Register) verdoppelt werden müssen.

Im zweiten Fall ist eine Freigabelogik der Transferrichtung dem bidirektionalen Draht so zugeordnet, daß ein wechselnder Transfer in beide Richtungen auf dem besagten Draht möglich ist. Diese Logik kann in den Verwaltungsprozessor ( $PG_j$ ), der dem Cache-Speicher ( $MC_j$ )

zugeordnet ist und mit dem der besagte bidirektionale Draht verbunden ist, integriert werden.

Selbstverständlich kann jede Serienverbindung gegebenenfalls mit einer höheren Anzahl von Seriendrähten durchgeföhrt werden.

Bei dem erfindungsgemäßen Mehrprozessorsystem können die Adressen-Übertragungsmittel im wesentlichen auf zwei Arten ausgeföhrt werden: zunächst können sie aus einem Bus für die parallele Übertragung von Blockadressen (BUSA) bestehen, der derselbe für alle Prozessoren (CPU<sub>j</sub>) ist und der diese letzteren und den Hauptspeicher (RAM) auf herkömmliche Art und Weise mit einem Busarbiter (AB) verbindet, der geeignet ist, die Zugriffskonflikte auf den besagten Bus zu verwalten. Es muß bemerkt werden, daß dieser Adreßbus nur für die Übertragung der Blockadressen benutzt wird: was die Struktur betrifft, so ist dieser Bus mit dem Bus für die parallele Adreßübertragung der bekannten Systeme, für den es deshalb keine Sättigungsprobleme gibt, da er sofort nach dem Transfer der Blockadresse freigemacht werden kann, identisch.

Es kann bei dem erfindungsgemäßen Mehrprozessorsystem jedoch noch eine andere Durchführungsart dieser Adressen-Übertragungsmittel ins Auge gefaßt werden; sie besteht darin, die Serienverbindungen für den Transfer der Informationsblöcke (bi) zu nutzen, um die Adressen dieser Blöcke zu übertragen.

In diesem Fall wird an jede Serienverbindung (LS<sub>j</sub>) parallel zum entsprechenden Speicher-Schieberegister (RDM<sub>j</sub>) ein zusätzliches Schieberegister (RDC<sub>j</sub>) angeschlossen: die durch die besagte Serienverbindung übertragenen Adressen werden so in jedes dieser Zusatzregister (RDC<sub>j</sub>) geladen; ein mit den besagten Registern (RDC<sub>j</sub>) und dem Hauptspeicher (RAM) verbundener Zugriffsverwaltungsarbiter (ABM) ist dann vorgesehen, um die in den besagten Registern befindlichen Adressen zu holen und die Zugriffskonflikte zum Hauptspeicher (RAM) zu verwalten. Ein solcher Arbiter ist von der Konzeption her

an sich bekannt, und diese Art von Zugriffskonflikten ist seit vielen Jahren gelöst. Bei dieser Durchführungsart wird zwar ein paralleler Adreßbus vermieden, jedoch sind die Verwaltungsbetriebsmittel mit mehr Aufwand verbunden.

5           Im übrigen ist das erfindungsgemäße Mehrprozessorsystem ganz besonders gut geeignet, um auf wirkungsvolle Weise die Kohärenzprobleme der zwischen Verarbeitungs-Prozessoren geteilten Daten zu verwalten. Tatsächlich haben die herkömmlichen Lösungen zur  
10 Verwaltung dieser geteilten Daten bei den bekannten Systemen aufgrund des Engpasses bei der Informationsübertragung ihre Grenzen, sind jedoch, im Gegenteil dazu, bei dem erfindungsgemäßen System, wo ein solcher Engpaß nicht mehr besteht, vollkommen befriedigend  
15 und leistungsfähig, sodaß dieses System mit Verwaltungsmitteln für die geteilten Daten ausgestattet werden kann, die mit denen der bekannten Systeme analog sind.

          Eine traditionelle Verwaltungslösung der  
20 geteilten Daten besteht zum Beispiel in der Vermeidung ihrer Übertragung durch die Cache-Speicher: auf herkömmliche Art und Weise ist eine Partitionslogik ( $LP_j$ ) jedem Verarbeitungs-Prozessor ( $CPU_j$ ) zugeordnet, um die Adressen der geteilten Daten und die der nicht geteilten  
25 Daten zu unterscheiden, und zwar, um die ersten direkt zum Hauptspeicher (RAM) und die zweiten zum entsprechenden Cache-Speicher ( $MC_j$ ) zu leiten.

          Bei einer ersten erfindungsgemäßen Architekturversion umfaßt das System:

30           . einen speziellen parallelen Wortübertragungsbus (BUSD), der die Prozessoren ( $CPU_j$ ) und den Hauptspeicher (RAM) verbindet,

          . eine jedem Prozessor ( $CPU_j$ ) zugeordnete Partitionslogik ( $LP_j$ ), die geeignet ist, die Adressen der  
35 geteilten Daten und die der nicht geteilten Daten zu unterscheiden, um diese auf den Adresen-Übertragungsmitteln mit ihrer Identifizierung zu übertragen,

eine dem Hauptspeicher (RAM) zugeordnete  
 Decodierungslogik (DEC), die geeignet ist, die Adressen  
 mit ihrer Identifizierung zu empfangen und die Daten am  
 Speicherausgang weiterzuleiten, entweder, was die  
 5 nichtgeteilten Daten betrifft, zum entsprechenden  
 Speicher-Schieberegister (RDM<sub>j</sub>) oder, was die geteilten  
 Daten betrifft, zu dem speziellen Wortübertragungsbuss  
 (BUSD).

Diese Lösung hat den Vorteil, daß sie  
 10 architekturenmäßig sehr einfach ist: das Vorhandensein des  
 speziellen parallelen Wortübertragungsbusses (BUSD) führt  
 zu besseren Leistungen im Vergleich zu einer Lösung, die  
 darin bestünde, die Serienverbindungen zu benutzen, um  
 nicht nur die Blöcke von nicht geteilten Daten, sondern  
 15 auch die Wörter von geteilten Daten zu übertragen. Es muß  
 angemerkt werden, daß diese letzte Lösung gegebenenfalls  
 bei einem schwachen Volumen von geteilten Daten ins Auge  
 gefaßt werden kann.

Bei einer anderen Version ist das System mit  
 20 einem speziellen parallelen Wortübertragungsbuss (BUSD) und  
 einem speziellen gemeinsamen Bus für die Übertragung der  
 Wortadressen (BUSAM) ausgestattet, die die Prozessoren  
 (CPU<sub>j</sub>) mit dem Hauptspeicher (RAM) verbinden. Die  
 Partitionslogik (LP<sub>j</sub>) leitet die Adressen der geteilten  
 25 Daten zu dem speziellen gemeinsamen Bus (BUSAM) für den  
 Datentransfer durch den speziellen Wortbus (BUSD) und  
 leitet die nicht geteilten Daten zu den Adressen-  
 Übertragungsmitteln (unabhängig davon, ob diese aus einem  
 parallelen Übertragungsbuss bestehen oder ob die  
 30 Übertragung über die Serienverbindungen erfolgt).

Das Vorhandensein eines speziellen Busses für  
 die Übertragung von Wortadressen ermöglicht bei dieser  
 Version, im Falle hoher Anforderungen nach geteilten  
 Daten, die Sättigungsgrenze der Adressen-  
 35 Übertragungsmittel herabzusetzen.

Eine andere Version, die in der Praxis dort  
 vorzuziehen ist, wo die Adressen-Übertragungsmittel aus  
 einem parallelen Adreßbus (BUSA) bestehen, besteht darin,

das System mit einem dem Speicher (RAM) zugeordneten Speicher-Verwaltungsprozessor (PGM) und mit einem jedem Verarbeitungs-Prozessor (CPU<sub>j</sub>) und dem entsprechenden Verwaltungsverzeichnis (RG<sub>j</sub>) zugeordneten Buspion-Prozessor (PE<sub>j</sub>) auszustatten; der Speicher-Verwaltungsprozessor (PGM) und jeder Buspion-Prozessor (PE<sub>j</sub>) - von der Struktur her an sich bekannt - sind an den Adreßbus (BUSA) angeschlossen, um die auf dem besagten Bus übertragenen Blockadressen zu überwachen beziehungsweise zu verarbeiten, und zwar so, daß ein Update des Hauptspeichers (RAM) und des zugeordneten Cache-Speichers (MC<sub>j</sub>) bei Entdecken einer in dem zugeordneten Verzeichnis (RG<sub>j</sub>) vorhandenen Blockadresse möglich ist.

Der Speicher-Verwaltungsprozessor (PGM) und jeder Spion-Prozessor (PE<sub>j</sub>) ordnen jedem Informationsblock Zustandsbits zu, aktualisieren diese je nach Art (Lesen oder Schreiben) der Blockanforderungen, die auf dem Bus (BUSA) übertragen werden, und sorgen für die Kohärenz der geteilten Daten, indem sie diese Zustandsbits benutzen, die es ihnen ermöglichen, das Schreiben eines Blocks im Moment der Anforderungen auf dem Bus (BUSA) in den Hauptspeicher zu erzwingen oder nicht.

In dem vorgenannten Fall, bei dem die Adressenübertragung über die Serienverbindungen erfolgt, können die geteilten Daten auch zentral verwaltet werden, und zwar durch einen dem Hauptspeicher (RAM) zugeordneten Speicher-Verwaltungsprozessor (PGM) und einen jedem Verarbeitungs-Prozessor (CPU<sub>j</sub>) und dem entsprechenden Verwaltungsverzeichnis (RG<sub>j</sub>) zugeordneten Prozessor zur Aufrechterhaltung der Kohärenz der geteilten Daten (PMC<sub>j</sub>), wobei jeder Prozessor zur Aufrechterhaltung der Kohärenz (PMC<sub>j</sub>) an einen Synchronisationsbus (SYNCHRO) angeschlossen ist, der von dem Speicher-Verwaltungsprozessor (PGM) so gesteuert wird, daß ein Update des Hauptspeichers (RAM) und des zugeordneten Cache-Speichers (MC<sub>j</sub>) bei Entdecken einer Blockadresse sowie ein Update des Hauptspeichers (RAM) und der Cache-

Speicher ( $MC_j$ ) bei jeder Entnahme von Adressen aus den Zusatzschieberegistern ( $RDC_j$ ) möglich ist.

Wie zuvor sorgen Zustandsbits, die durch den Prozessor (PGM) jedem Informationsblock zugeordnet sind, für dieses Update.

Es ist anzumerken, daß in der vorhergehenden Architektur, bei der die Blockadressen auf einem gemeinsamen Adreßbus (BUSA) übertragen werden, gegebenenfalls ein Synchronisationsbus der oben definierten Art vorgesehen werden kann. In diesem Fall werden die Spionprozessoren ( $PE_j$ ) durch den Speicher-Verwaltungsprozessor (PGM) über den Synchronisationsbus beansprucht, dies jedoch nur dann, wenn sie von dem Transfer betroffen sind. So werden überflüssige Zugriffe auf die Cache-Speicher vermieden; die Spionprozessoren werden dann passiv (da sie von dem Prozessor PGM beansprucht werden) und sie werden daher gemäß der oben benutzten Terminologie besser durch den angemesseneren Ausdruck "Prozessor zur Aufrechterhaltung der Kohärenz" bezeichnet.

Eine andere Lösung besteht darin, den parallelen Adreßbus (BUSA) dem Transfer der Blockadressen von geteilten Daten vorzubehalten und die Serienverbindungen für den Transfer der Blöcke der nicht geteilten Daten zu benutzen.

Im übrigen ist das erfindungsgemäße Mehrprozessorsystem für Gruppen von Verarbeitungs-Prozessoren auf ein und derselben Serienverbindung geeignet, sodaß die Anzahl der notwendigen Serienverbindungen und die der entsprechenden Speicher-Schieberegister ( $RDM_j$ ) beschränkt wird.

Die Anzahl der Speicher-Schieberegister ( $RDM_j$ ) kann der Anzahl der Serienverbindungen ( $LS_j$ ) entsprechen; in diesem Fall wird jedes Speicher-Schieberegister ( $RDM_j$ ) statisch an eine Serienverbindung ( $LS_j$ ), die spezifisch zu dem besagten Register gehört, angeschlossen.

Die Anzahl der Speicher-Schieberegister ( $RDM_j$ ) kann auch von der der Serienverbindungen ( $LS_j$ )

verschieden, insbesondere niedriger sein; in diesem Fall werden diese Register durch ein Verbundnetz dynamisch an die Serienverbindungen ( $LS_j$ ) angeschlossen.

Wie bei den herkömmlichen Systemen kann der  
 5 Hauptspeicher (RAM) in  $m$  parallel angeordnete Speicherbanken ( $RAM_1 \dots RAM_p \dots RAM_m$ ) unterteilt werden. Jedes Speicher-Schieberegister ( $RDM_j$ ) besteht dann aus  $m$  elementaren Registern ( $RDM_{j1} \dots RDM_{jp} \dots RDM_{jm}$ ), die parallel mit der entsprechenden Serienverbindung ( $LS_j$ )  
 10 verbunden sind. Jedoch wird ein zusätzlicher Parallelitätsgrad und eine bessere elektrische oder optische Anpassung der Verbindung bei einer Variante erreicht, bei der jede Speicherbank  $RAM_p$  durch eine Punkt zu Punkt-Serienverbindung  $LS_{jp}$  mit jedem Prozessor  $CPU_j$   
 15 verbunden ist.

Im übrigen wird, um eine Transferleistung zu erreichen, die wenigstens genauso groß ist, wie die der herkömmlichen Systeme mit Parallelbus, das erfindungsgemäße System vorzugsweise durch einen Taktgeber  
 20 synchronisiert, dessen Frequenz  $F$  wenigstens 100 Megahertz beträgt. Die Speicher-Schieberegister ( $RDM_j$ ) und Prozessor-Schieberegister ( $RDP_j$ ) können ganz einfach von einem Typ sein, der für eine Schiebefrequenz von wenigstens  $F$  geeignet ist.

25 Im Fall sehr hoher Frequenzen (bei der aktuellen Technologie insbesondere höher als 500 Megahertz) können diese Register in Unterregister von einer schwächeren Schiebefrequenz unterteilt und multiplexiert werden.

30 Die Erfindung erstreckt sich weiterhin auf ein Serienmultiport-Speicherbauteil, das dazu dient, das zuvor definierte Mehrprozessorsystem auszurüsten, um deren Herstellung zu vereinfachen. Dieses Bauteil, das im übrigen verschiedene Anwendungen haben kann, besteht aus  
 35 einem integrierten Schaltkreis, der einen Speicher mit Zufallsreihenfolgezugriff (RAM) von einer zuvor festgelegten Größe umfaßt, die einem Informationsblock ( $bi$ ) entspricht, mehreren Schieberegistern ( $RDM_1 \dots RDM_j$

...  $RDM_n$ ), wobei jedes Register eine der Größe des Speichers entsprechende Kapazität aufweist, einem internen parallelen Bus (BUSI), der den Zugang des Speichers und die Schieberegister verbindet, einer Logik zur Auswahl  
 5 eines Schieberegisters (LSR), die geeignet ist, die Verbindung auf dem internen Bus zwischen dem Speicher und einem zuvor bestimmten Schieberegister freizugeben und mehreren externen Eingangs-/Ausgangsstiften für die Adresseneingabe in den Speicher (RAM), für die  
 10 Adresseneingabe zu der Auswahllogik (LSR), für die Eingabe und die Freigabe von Transferbefehlen zum Lesen oder Schreiben eines Informationsblocks (bi) zwischen dem Speicher (RAM) und den Schieberegistern ( $RDM_j$ ), für die Eingabe eines Taktgebersignals zu jedem Schieberegister  
 15 ( $RDM_j$ ), für die bit-für-bit-Eingabe eines Informationsblocks (bi) zu jedem Schieberegister ( $RDM_j$ ) und für die bit-für-bit-Ausgabe eines Informationsblocks aus jedem Schieberegister ( $RDM_j$ ).

Dieses Bauteil kann durch Hinzufügen von Konfigurationsregistern ( $RC_1$ ,  $RC_2$ , ...) parametrierbar  
 20 gemacht werden, die insbesondere die Wahl der Größe der Informationsblöcke (bi) und verschiedener Betriebsmodi der Schieberegister ermöglichen.

Die in ihrer allgemeinen Form dargestellte  
 25 Erfindung wird durch die folgende Beschreibung illustriert, die sich auf die im Anhang befindlichen Zeichnungen bezieht; diese Zeichnungen stellen in einer nicht beschränkten Weise mehrere Durchführungsarten dar; auf diesen Zeichnungen, die einen wesentlichen Teil der  
 30 vorliegenden Beschreibung darstellen:

- ist Abbildung 1 ein synoptisches Schema einer ersten Durchführungsart des erfindungsgemäßen Mehrprozessorsystems,

- ist Abbildung 2 ein Diagramm, das die  
 35 errechnete Leistungskurve dieses Systems (A) zeigt und zum Vergleich dazu die entsprechende Kurve (B) für eine konventionelle Mehrprozessor-Architektur mit einem gemeinsamen Bus,

- sind die Abbildungen 3, 4 und 5 detaillierte logische Schemata funktioneller Einheiten des Systems der Abbildung 1,

5 - ist Abbildung 6 ein synoptisches Schema einer anderen Durchführungsart des Systems,

- ist Abbildung 7 ein synoptisches Schema eines Systems desselben Typs, wie in Abbildung 1, das mit Verwaltungsmitteln der geteilten Daten ausgestattet ist,

10 - ist Abbildung 8 ein logisches Schema, ein Detail einer Unterbaugruppe des Systems der Abbildung 7,

- ist Abbildung 9 ein synoptisches Schema eines zu Abbildung 7 analogen Systems mit einer Variante bezüglich der Verwaltungsmittel von geteilten Daten,

15 - ist Abbildung 10 ein synoptisches Schema eines analogen Systems, das mit unterschiedlichen Verwaltungsmitteln von geteilten Daten ausgestattet ist,

20 - sind die Abbildungen 11, 12a, 12b, 12c, 12d, 13, 14, 15, 16 und 17 detaillierte logische Schemata funktioneller Einheiten des Prozessorsystems der Abbildung 10,

- ist Abbildung 18 ein synoptisches Schema eines Systems desselben Typs, wie in Abbildung 6, das mit Verwaltungsmitteln von geteilten Daten ausgestattet ist,

25 - ist Abbildung 19 ein vereinfachtes synoptisches Schema einer Systemvariante, bei der sich mehrere Zentraleinheiten denselben Seriendraht teilen,

- ist Abbildung 20a ein synoptisches Schema einer bevorzugten Durchführungsart, bei der der Hauptspeicher in mehrere Speicherbanken angeordnet ist,

30 - ist Abbildung 20b eine Variante der in Abbildung 20a dargestellten Architektur,

- schematisieren die Abbildungen 21a und 21b eine andere RAM-Speicherstruktur, mit der das besagte System ausgerüstet werden kann,

35 - ist Abbildung 22 ein synoptisches Schema, das die Struktur eines Serienmultiport-Speicherbauteils darstellt, mit dem das System ausgerüstet werden kann.

Die in Abbildung 1 in synoptischer Form dargestellte Vorrichtung ist ein Mehrprozessorsystem, das  $n$  Verarbeitungs-Prozessoren  $CPU_1 \dots CPU_j \dots CPU_n$  besitzt. In dieser Abbildung sind zwei Verarbeitungs-Prozessoren  $CPU_1$  und  $CPU_j$  mit ihrer zugeordneten Logik dargestellt. Beide Verarbeitungs-Prozessoren sind herkömmlicher Art, zum Beispiel "MOTOROLA 68020" oder "INTEL 80386" ... und können lokale Speicher- und periphere Schnittstellenhilfsmittel haben und mit einer virtuellen Speichervorrichtung ausgestattet werden.

Die Vorrichtung umfaßt einen Hauptspeicher mit Zufallsreihenfolgezugriff RAM, der in herkömmlicher Art mit integrierten Speicherschaltkreisen ausgeführt ist: insbesondere "INTEL" "NEC" "TOSHIBA" ... dynamischer RAM mit 256 Kbit, 1 Mbit, 4 Mbit ... je nach Anwendung. Dieser Speicher ist in Informationsblöcken  $b_0 \dots b_i \dots$  mit der festgelegten Größe  $t$  angeordnet (normalerweise 256 Bit bis 2 Kbit), und die Zugriffsfront des besagten Speichers entspricht der Größe eines Blocks.

Der Hauptspeicher ist parallel mit  $n$  Schieberegistern  $RDM_1 \dots RDM_j \dots RDM_n$ , sogenannten Speicherregistern, verbunden, wobei jedes Speicherregister die Größe  $t$  eines Informationsblocks hat; jedes dieser Register ist in einer sehr schnellen Technologie ("ASGA") ausgeführt, und ein Block kann innerhalb eines Zyklus des Hauptspeichers RAM geladen und entladen werden. Die Anzahl  $n$  der Register ist gleich der Anzahl der Prozessoren  $CPU_j$ .

Im übrigen ist ein Cache-Speicher  $MC_j$  in einer an sich bekannten Art jedem Prozessor  $CPU_j$  zugeordnet; jeder Cache-Speicher besteht nach herkömmlicher Art aus einem Speicher mit schnellem Zugriff und Zufallsreihenfolgezugriff von einer im Verhältnis zum Hauptspeicher RAM schwachen Kapazität. Ein Verzeichnis  $RG_j$  und ein Verwaltungsprozessor  $PG_j$  sind nach traditioneller Art mit dem Cache-Speicher und mit dem Verarbeitungs-Prozessor verbunden, um die Informationen, die durch den Cache-Speicher hindurch übertragen werden, zu verwalten.

Außerdem ist in dem erfindungsgemäßen System ein Schieberegister  $RDP_j$ , ein sogenanntes Prozessorregister, durch seinen parallelen Port mit jedem Cache-Speicher  $MC_j$  verbunden; die Größe von jedem  
 5 Prozessorregister  $RDP_j$  entspricht der Größe eines Blocks  $b_i$  und die Struktur ist ähnlich der eines Speicherregisters  $RDM_j$ .

Jedes Speicherregister  $RDM_j$  ist durch seinen Serienport mit dem Serienport eines Prozessorregisters  
 10  $RDP_j$  durch eine Serienverbindung  $LS_j$  verbunden. Durchführungsbeispiele dieser Serienverbindung, die einen bidirektionalen Draht oder zwei unidirektionale Drähte umfassen kann, sind in den Abbildungen 4 und 5 dargestellt. Die Transferkontrolle der Blöcke  $b_i$  zwischen  
 15 entsprechenden Registern  $RDM_j$  und  $RDP_j$  wird durch Transferlogiken  $TFR_j$  und  $TFR'_j$  durchgeführt, die dem Speicherregister  $RDM_j$  und dem Prozessorregister  $RDP_j$  symmetrisch zugeordnet sind; ein Durchführungsbeispiel dieser Transferlogiken (an sich herkömmlicher Art) wird in  
 20 Abbildung 3 detailliert dargestellt.

Der Hauptspeicher RAM, die Speicher-Schieberegister  $RDM_1 \dots RDM_n$  und die zugeordneten Transferlogiken  $TFR_1 \dots TFR_n$  bilden zusammen eine funktionelle Einheit, "Serienmultiport-Speicher" MMS  
 25 genannt. Der Verarbeitungs-Prozessor  $CPU_j$ , der Cache-Speicher  $MC_j$ , das Verwaltungsverzeichnis des Cache-Speichers  $RG_j$ , der Verwaltungsprozessor des Cache-Speichers  $PG_j$ , das Prozessor-Schieberegister  $RDP_j$  und die zugeordnete Transferlogik  $TFR'_j$  bilden zusammen eine  
 30 "funktionelle" Einheit, "Zentraleinheit"  $UC_j$  genannt.

Im übrigen umfaßt das System Mittel für die Übertragung von Blockadressen von den Prozessoren  $CPU_j$  zum Hauptspeicher RAM, die in dem Beispiel aus einem gemeinsamen parallelen Bus  $BUSA$  bestehen, an dem die  
 35 Prozessoren  $CPU_j$  (durch ihren Verwaltungsprozessor  $PG_j$ ) und der Hauptspeicher RAM angeschlossen werden.

Der Zugriff auf den Bus BUSA wird auf herkömmliche Art und Weise durch einen Busarbiter AB verwaltet.

Die oben definierte Architektur funktioniert  
5 folgendermaßen:

Ein Prozessor CPU<sub>j</sub> führt ein eigenes Programm aus, das aus Befehlen besteht, die sich in Wortform im Hauptspeicher RAM befinden mit Auszügen in dem zugeordneten Cache-Speicher MC<sub>j</sub>. Auf die Programmbefehle  
10 hin liest der Prozessor CPU<sub>j</sub> entweder Datenwörter, wobei sich die Daten selbst im Hauptspeicher RAM oder im Cache-Speicher MC<sub>j</sub> in Form von Auszügen befinden, oder schreibt Datenwörter in den Hauptspeicher RAM und in den Cache-Speicher MC<sub>j</sub>.

15 Für jede Operation eines Prozessors CPU<sub>j</sub> (als "Anforderung" bezeichnet) muß die Adresse adr des betreffenden Worts, die Art, r oder w der Operation (Lesen, Schreiben) und der Austausch (data) des betreffenden Worts geliefert werden.

20 Jede Wort-Anforderung aktiviert den Prozessor PG<sub>j</sub>, der dann auf herkömmliche Art und Weise das Verzeichnis des Cache-Speichers RG<sub>j</sub> konsultiert, das angibt, ob der Block bi, der das betreffende Wort enthält, im Cache-Speicher MC<sub>j</sub> vorhanden ist und gegebenenfalls den  
25 Bereich des Blocks im Cache-Speicher, wo sich der gesuchte Block befindet.

Befindet sich der Block bi, der das betreffende Wort enthält, im Cache-Speicher MC<sub>j</sub>, dann wird bei dem Lesevorgang dieses Wort im besagten Cache-Speicher  
30 gelesen und zum Prozessor CPU<sub>j</sub> gesandt; im Fall des Schreibens wird das vom Prozessor CPU<sub>j</sub> gelieferte Wort in den Cache-Speicher geschrieben: die Speichertransaktion ist beendet.

Befindet sich der Block, der das betreffende  
35 Wort enthält, nicht im Cache-Speicher MC<sub>j</sub>, dann ist ein Lesen des Blocks bi im Hauptspeicher RAM notwendig. Es kann zwei Fälle geben.

Erster Fall

Der Cache-Speicher  $MC_j$  verfügt über wenigstens eine freie Blockstelle, die vom Prozessor  $PG_j$  mittels Zustandsbits, die jeder Eingabe des Verzeichnisses  $RG_j$  zugeordnet sind, bestimmt wird. In diesem Fall fragt der

5 Prozessor  $PG_j$  auf herkömmliche Art und Weise den Bus  $BUSA$  ab, indem er dem Busarbiter  $AB$  seine Anforderung vorlegt. Dieser letztere gibt, wenn er an der Reihe ist, den Bus  $BUSA$  zum Prozessor  $PG_j$ , der für den Lesevorgang auf den Hauptspeicher  $RAM$  Zugriff nimmt, und der im Speicher

10 gelesene Block wird in das Register  $RDM_j$  geladen, das durch die Nummer  $j$  des Abfragers identifiziert wird. Das Ende des Lesezyklus hat das Freimachen des Busses  $BUSA$  und die Aktivierung des Transfers mit der Serienverbindung  $LS_j$ , was den Transfer des Inhalts des Speicherregisters

15  $RDM_j$  in das Prozessorregister  $RDP_j$  ermöglicht, zur Folge. Das Ende des Transfers aktiviert das Schreiben des Inhalts des Prozessorregisters in den Cache-Speicher  $MC_j$ , und zwar an der Blockstelle, die diesem Zweck vorbehalten ist, und die Transaktion kann wie zuvor beendet werden.

#### 20 Zweiter Fall

Verfügt der Cache-Speicher  $MC_j$  über keine freie Stelle, dann wird durch einen klassischen Algorithmus eine Stelle im Cache-Speicher dazu bestimmt, den angeforderten Block zu empfangen. Zwei Situationen

25 können auftreten:

- der sich an der bestimmten Stelle befindliche Block ist seit seiner Installierung nicht modifiziert worden: er wird einfach dadurch eliminiert, daß der Bereich des Blocks durch einfaches Schreiben eines

30 Zustandsbits in das Verzeichnis ( $RG_j$ ) freigemacht wird, und dann kann die Transaktion wie zuvor fortgeführt werden,

- der sich an der bestimmten Stelle befindliche Block ist modifiziert worden, und es ist ein

35 Update des Hauptspeichers  $RAM$  notwendig. Dazu transferiert der Verwaltungsprozessor  $PG_j$  den bestimmten Block in das Prozessorregister  $RDP_j$ , aktiviert den Transfer vom Prozessorregister  $RDP_j$  in das Speicherregister  $RDM_j$  und

fragt dann den gemeinsamen Bus BUSA ab, indem er dem Busarbitrator AB seine Anforderung vorlegt. Gibt der Arbitrator den Bus dem Verwaltungsprozessor PG<sub>j</sub>, dann aktiviert dieser letztere einen Schreibbefehl, wodurch der Inhalt  
 5 des Speicherregisters RDM<sub>j</sub> an seine Stelle im Hauptspeicher RAM übertragen wird. Das Update des RAM-Speichers ist beendet, und die Transaktion kann wie zuvor weitergeführt werden.

So erfolgt der Austausch zwischen den  
 10 Verarbeitungs-Prozessoren CPU<sub>j</sub> und ihrem Cache-Speicher MC<sub>j</sub> und zugeordneten Logiken RG<sub>j</sub> und PG<sub>j</sub> bei der erfindungsgemäßen Vorrichtung, auf herkömmliche Art und Weise; im Gegensatz dazu erfolgen die Blocktransfers zwischen Hauptspeicher RAM und Cache-Speichern MC<sub>j</sub> nicht  
 15 mehr über einen gemeinsamen parallelen Bus, sondern über Serienverbindungen LS<sub>j</sub>, die für jeden Verarbeitungs-Prozessor CPU<sub>j</sub> bestimmt sind, der gemeinsame Bus BUSA dient nur zum Transfer der Adressen und hat daher einen erheblich niedrigeren Verkehr.

Es ist bekannt, daß für die herkömmlichen Architekturen mit einem gemeinsamen Bus eine Modellierung, die von "PATEL" untersucht worden ist, (analysis of multiprocessors with private cache, JANAK H. PATEL - IEEE Transactions on computers vol. C. 31, N° 4 APRIL 1982) zu  
 25 der annähernden folgenden Formel geführt hat, die abhängig von der Anzahl der vorhandenen Prozessoren den Wirkungsgrad U angibt:

$$U = 1 / 1 + m (W + tf)$$

30

wobei

der Wirkungsgrad U die mittlere Nutzungsrate jedes Verarbeitungs-Prozessors ist,

m die Wahrscheinlichkeit ist, daß ein Verarbeitungs-  
 35 Prozessor Daten abfragt, die nicht in seinem Cache-Speicher vorhanden sind (diese Wahrscheinlichkeit m = alpha.Pa verhält sich proportional zu der Wahrscheinlichkeit des Nichtvorhandenseins Pa der

Information in dem Cache-Speicher und zu einem Faktor  $\alpha$ , der abhängig ist von der Leistung des Verarbeitungs-Prozessors, der auf einen Prozentsatz von Speicherabfragen gebracht worden ist),

- 5 W die durchschnittliche Wartezeit des gemeinsamen Busses ist, abhängig von der Anzahl  $n$  der Prozessoren,  $t_f$  die Transferdauer eines Blocks vom Hauptspeicher in einen Cache-Speicher ist.

Die Annahmen, von denen ausgehend diese Formel  
10 aufgestellt worden ist, zeigen, daß diese auf die erfindungsgemäße Architektur anwendbar ist, mit einem Approximationsgrad, der mit dem Approximationsgrad der Formel für klassische Architekturen mit einem gemeinsamen Bus vergleichbar ist.

15 Es ist somit möglich, die Leistungen der beiden Architekturarten zu vergleichen, unter der Voraussetzung, daß die Bauteile, die beide Architekturen gemein haben, identische Eigenschaften haben.

Abbildung 2 zeigt die Kurven, die man aus dem  
20 von der Anzahl  $n$  der Prozessoren abhängigen Wirkungsgrad  $U$  für die folgenden Parameter erhält, wobei die Parameter, die beide Vorrichtungen gemein haben, identisch sind und alle übliche Werte haben:

- Größe des Blocks  $b_i = 64$  Byte,
- 25 - Größe des Worts für den Paralleltransfer auf dem gemeinsamen Bus = 4 Byte,
- Zugriffsdauer zum Hauptspeicher RAM = 100 Nanosekunden,
- Zyklusdauer des Busses BUSA = 50 Nanosekunden,
- Frequenz des Serientransfers = 500 MHz,
- 30 - Wahrscheinlichkeit des Nichtvorhandenseins  $P_a = 0,005$  (Cache-Speicher von 16 KByte),
- Leistungsfaktor der Prozessoren:  $\alpha = 0,5$ .

Beim Vergleich der Kurven A (erfindungsgemäße Architektur) und B (herkömmliche Architektur) stellt man  
35 fest, daß die erfindungsgemäße Architektur einen deutlich höheren Wirkungsgrad, als die herkömmliche Architektur hat; die erfindungsgemäße Architektur ermöglicht den Einsatz einer sehr viel höheren Anzahl von Prozessoren als

die herkömmlichen Architekturen mit einem gemeinsamen Bus, die in der Praxis circa zehn Prozessoren nicht überschreiten können. So wird zum Beispiel in dem herkömmlichen Fall ein Wirkungsgrad von 0,75 schon bei dem  
 5 zehnten Prozessor erreicht, wohingegen er für mehr als 80 Prozessoren im Falle der Erfindung erreicht wird.

Abbildung 3 stellt eine Durchführungsart einer Transferlogik  $TFR_j$  oder  $TFR'_j$  dar, die den Transfer eines Informationsblocks  $bi$  von einem Speicherregister  $RDM_j$  zu  
 10 einem Prozessorregister  $RDP_j$  ermöglicht (für den Transfer in die andere Richtung sorgen symmetrische Mittel, die in dieser Abbildung nicht dargestellt werden). Jede Logik  $TFR_j$  oder  $TFR'_j$  enthält einen Sendekontrollteil  $TFRE_j$  und  $TFRE'_j$  und einen Empfangskontrollteil  $TFRR_j$  und  $TFRR'_j$ ,  
 15 die gekreuzt aktiviert werden (Senden  $TFRE_j$  wird synchron mit dem Empfang  $TFRR'_j$  aktiviert). Das System weist einen Taktgebergenerator  $H$  auf, dessen Frequenz die Übertragungsgeschwindigkeit festlegt und dem Sendeteil  $TFRE_j$  und dem Empfangsteil  $TFRR'_j$  das Taktgebersignal  $h$   
 20 liefert.

In dem Sendeteil  $TFRE_j$  ermöglicht ein Dekrementalzähler  $DC$ , der durch seinen Ladeeingang  $load_2$  das Lesesignal  $\bar{r}$  des Verwaltungsprozessors  $PG_j$  erhält, das Durchlassen von  $t + 1$  Taktgeberimpulsen  $h$  durch eine  
 25 Verknüpfung  $ET1$  hindurch, die durch ein Null-Durchgangssignal auf "borrow" gesteuert wird, wobei der Ausgang dieser Verknüpfung  $ET1$  mit dem Dekrementiereingang "down" des Dekrementalzählers  $DC$  und mit dem Schiebeingang  $shift1$  des Speicherregisters  $RDM_j$  verbunden ist.

In dem Empfangsteil  $TFRR'_j$  ist eine Kippschaltung  $B$  durch ihren Dateneingang  $D$  mit dem Serienausgang des Prozessorregisters  $RDP_j$  verbunden, und der Taktgebereingang  $clk$  dieser Kippschaltung ist mit dem Taktgeber  $H$  verbunden, um das Signal  $h$  zu empfangen. Ein  
 30 vom Verwaltungsprozessor  $PG_j$  geliefertes Initialisierungssignal "init" ist mit dem Eingang  $S$  der Kippschaltung  $B$  und dem Ladeeingang  $load3$  des Prozessorregisters  $RDP_j$  verbunden. Der Ausgang  $Q$  der

Kippschaltung sendet an die Verknüpfung ET2 ein Steuersignal fin\_transfer, das das Durchlassen des Taktgebersignals  $h$  zum Schiebееingang  $shift2$  des Prozessorregisters  $RDP_j$  ermöglicht. Dieses Steuersignal  
 5 wird ebenfalls an den Verwaltungsprozessor  $PG_j$  ausgegeben, um das Ende des Blocktransfers anzuzeigen.

Das Ganze funktioniert wie folgt: der Verwaltungsprozessor  $PG_j$ , führt, nachdem er den Zugriff auf den Hauptspeicher RAM über den Bus BUSA erhalten hat,  
 10 seinen Lesevorgang des Blocks  $bi$  aus, indem er die Adresse des betreffenden Blocks und das Lesesignal  $\bar{r}$  liefert. Dieses Signal löst die Aktivierung des Sendeteils  $TFRE_j$  aus: die letzte Flanke des Lesesignals  $r$  bewirkt das Laden des Blocks  $bi$  im Speicherregister  $RDM_j$ , durch die  
 15 Aktivierung des Signals load1 und das Laden des  $t + 1$ -Wertes, der der Größe des Blocks  $bi$  in Bits plus einem zusätzlichem Bit, "start"-Bit genannt, entspricht, in den Dekrementalzähler DC durch die Aktivierung des Signals load2; dadurch wird das Signal borrow auf 1 zurückgesetzt  
 20 und der Transfertaktgeber H erhält die Erlaubnis, über die Verknüpfung ET1, die durch dieses borrow-Signal bedingt ist,  $t + 1$  Taktgeberimpulse  $h$  zu liefern: diese Impulse verschieben  $t + 1$  Bits des Speicherregisters  $RDM_j$  durch den  $shift1$ -Eingang und lassen den Wert 0 durch den Eingang  
 25 down des Dekrementalzählers DC erreichen: das Signal borrow wird auf Null zurückgesetzt und blockiert den Betrieb der Sendeteils  $TFRE_j$ .

So überträgt die Serienverbindung  $LS_j$ , zu Beginn im Logikruhezustand 1, das Bit 0, das sogenannte  
 30 Start-Bit, dann die  $t$  Bits des Blocks  $bi$  und geht dann wieder in den Logikruhezustand 1, wobei das zuletzt gesandte Bit der auf den Serieneingang des Speicherregisters  $RDM_j$  gesetzte Wert 1 ist.

Bevor der Lesevorgang angefordert wird, hat  
 35 der Verwaltungsprozessor  $PG_j$  den Empfangsteil  $TFRR'_j$  durch Aktivierung des Signals init initialisiert, wodurch das Prozessorregister  $RDP_j$  durch den Eingang load3 mit  $t$  Bits auf 1 geladen wird und der Ausgang Q der Kippschaltung B

durch den Eingang  $\bar{S}$  in den Logikzustand 1 gesetzt wird. Dieser Ausgang Q gibt dann die Verknüpfung ET2 frei, die das Taktgebersignal h zum Eingang shift2 des Prozessorregisters RDP<sub>j</sub> durchläßt. Bei jedem

5 Taktgeberimpuls liefert dieses Prozessorregister ein Bit, das in der Kippschaltung B gespeichert wird an seinen Serienausgang. Durch das erste ankommende 0-Bit wird der Ausgang Q der Kippschaltung B auf Null gesetzt und das Taktgebersignal h auf der Verknüpfung ET2 blockiert. Da

10 dieses erste 0-Bit das Start-Bit ist, das dem Block bi vorangeht, ist dieser letztere also im Prozessorregister RDP<sub>j</sub> "gefangen", während der Verwaltungsprozessor PG<sub>j</sub> von der Zustandsänderung der Kippschaltung B durch das Signal fin\_transfer informiert wird: der Verwaltungsprozessor PG<sub>j</sub>

15 braucht diesen Block bi auf dem Parallelausgang des Registers RDP<sub>j</sub> nur noch zu lesen.

Das Schreiben eines Blocks bi in den Hauptspeicher RAM benötigt das Vorhandensein einer Logik TFRE'<sub>j</sub>, die mit der Logik TFRE<sub>j</sub> identisch und dem

20 Prozessorregister RDP<sub>j</sub> zugeordnet ist, und einer Logik TFRR'<sub>j</sub>, die mit der Logik TFRR<sub>j</sub> identisch und dem Speicherregister RDM<sub>j</sub> zugeordnet ist. In diesem Fall ist das Signal init der Logik TFRR'<sub>j</sub> mit dem Schreibsignal w verbunden: das Freimachen des Speicherregisters RDM<sub>j</sub> setzt

25 automatisch die Empfangslogik TFRR<sub>j</sub> zurück.

Diese Ausführungsart der Logik zur Transferkontrolle ist nur ein mögliches Beispiel: das Senderegister kann selbst auch ständig verschoben sein und das Empfangsregister bei Entdecken des Start-Bits zu

30 Beginn des Transfers für t Taktgeberimpulse aktiviert sein.

Die Taktgeber H kann an beide Register angeschlossen sein, oder es können zwei unabhängige lokale Taktgeber benutzt werden, wobei die Synchronisation auf

35 herkömmliche Art und Weise durch eine sogenannte Synchronisationspräambel erreicht wird.

Das in Abbildung 4 dargestellte System umfaßt ein verdoppeltes Speicher-Schieberegister RDM1<sub>j</sub> und RDM2<sub>j</sub>,

ein verdoppeltes Prozessor-Schieberegister RDP1<sub>j</sub> und RDP2<sub>j</sub>, zwei unidirektionale Seriendrähte LS1<sub>j</sub> und LS2<sub>j</sub>, wobei der eine das Speicherregister RDM1<sub>j</sub> mit dem Prozessorregister RDP1<sub>j</sub> so verbindet, daß der Inhalt des  
 5 ersten zum zweiten übertragen wird und der andere das Speicherregister RDM2<sub>j</sub> mit dem Prozessorregister RDP2<sub>j</sub> so verbindet, daß der Inhalt des zweiten zum ersten übertragen wird, und es umfaßt zugeordnete Logiken zur Transferkontrolle: TFRE1<sub>j</sub> für RDM1<sub>j</sub>, TFRR2<sub>j</sub> für RDM2<sub>j</sub>,  
 10 TFRE2<sub>j</sub> für RDP2<sub>j</sub>, TFRR1<sub>j</sub> für RDP1<sub>j</sub>.

Um einen Informationsblock bi im Hauptspeicher RAM zu lesen, initialisiert der Verwaltungsprozessor PG<sub>j</sub> durch das Signal init die Logik TFRR1<sub>j</sub>, die dem Prozessorregister RDP1<sub>j</sub> zugeordnet ist, und aktiviert dann  
 15 durch das Lesesignal  $\bar{r}$  seine Lese-Anforderung an den RAM-Speicher. Dieses Signal aktiviert die dem Speicherregister RDM1<sub>j</sub> zugeordnete Logik TFRE1<sub>j</sub>: diese sorgt für den Transfer des Informationsblocks bi auf dem Draht LS1<sub>j</sub>. Das Ende des Transfers wird von der Logik TFRR1<sub>j</sub> entdeckt, die  
 20 dem Prozessorregister RDP1<sub>j</sub> zugeordnet ist, der den Verwaltungsprozessor PG<sub>j</sub> von der Ankunft des Blocks bi durch das Signal fin\_transfer informiert. Der Verwaltungsprozessor PG<sub>j</sub> überträgt dann den Inhalt des Prozessorregisters RDP1<sub>j</sub> in den Cache-Speicher MC<sub>j</sub>.

25 Um einen Speicherblock bi zu schreiben, lädt der Verwaltungsprozessor PG<sub>j</sub> das Prozessorregister RDP2<sub>j</sub> mit dem betreffenden Block bi, der aus dem Cache-Speicher MC<sub>j</sub> ausgezogen wird, was den Transfer dieses Blocks auf dem Draht LS2<sub>j</sub> aktiviert. Die dem Speicherregister RDM2<sub>j</sub> zugeordnete Transferlogik TFRR2<sub>j</sub> sorgt für den richtigen  
 30 Empfang dieses Blocks. Der Verwaltungsprozessor PG<sub>j</sub> wird durch die Zustandsänderung des Signals borrow, das aus der Übertragungslogik TFRE2<sub>j</sub> kommt, vom Ende des Transfers informiert. Der Verwaltungsprozessor PG<sub>j</sub> führt dann seine  
 35 Schreibanforderung aus, die bei Aktivierung des Schreibsignals  $\bar{w}$  effektiv wird: dadurch wird der Inhalt des Registers RDM2<sub>j</sub> in den Hauptspeicher RAM übertragen

und die Logik TFRR2<sub>j</sub> für einen nächsten Transfer erneut initialisiert.

Diese Vorrichtung erlaubt einen gleichzeitigen Transfer von Blöcken in beide Richtungen und ermöglicht  
 5 die schnellere Verarbeitung des Fehlens von Blöcken bei im Cache-Speicher MC<sub>j</sub>, wenn dieser letztere gesättigt ist; sie erlaubt auch den Einsatz eines herkömmlichen Antizipationsmechanismus zum Lesen von Blöcken.

In einer anderen in Abbildung 5 dargestellten Durchführungsart umfaßt die Verbindung LS<sub>j</sub> einen einzigen  
 10 bidirektionalen Draht, der an jedem Ende mit einer Freigabelogik LV<sub>1</sub> und LV<sub>2</sub> ausgestattet ist, die aus einer Verknüpfung mit zwei Eingängen mit offenem Kollektor OC<sub>1</sub> und OC<sub>2</sub> besteht, wobei einer der Eingänge mit dem  
 15 Serienausgang des Speicherregisters RDM<sub>j</sub> für die Verknüpfung OC<sub>1</sub> und des Prozessorregisters RDP<sub>j</sub> für die Verknüpfung OC<sub>2</sub> verbunden ist und der andere Eingang mit dem Ausgang Q einer Steuer-Kippschaltung BC<sub>1</sub> und BC<sub>2</sub> verbunden ist; jede dieser Kippschaltungen ist durch ihre  
 20 Eingänge  $\bar{S}$  und  $\bar{R}$  mit der Transferlogik TFR für die Kippschaltung BC<sub>1</sub> und TFR' für die Kippschaltung BC<sub>2</sub> verbunden.

Lesen und Schreiben erfolgt ausschließlich und wird nur vom Verwaltungsprozessor PG<sub>j</sub> aktiviert.

25 Das Lesen des Speichers aktiviert das Lesesignal  $\bar{r}$ , das durch ihren Eingang  $\bar{S}$  das Setzen der Kippschaltung BC<sub>1</sub> auf 1 bewirkt; das Zurücksetzen auf Null wird durch die Transferlogik TFR am Ende des Blocktransfers auf dem Eingang  $\bar{R}$  gesteuert.

30 Das Schreiben in einen Speicher löst denselben Mechanismus auf die Freigabelogik LV<sub>2</sub> aus.

Andere Kombinationen Register/Drähte sind möglich, und im Falle eines bidirektionalen Drahts können insbesondere bidirektionale Schieberegister benutzt  
 35 werden, die ein Signal für die Transferrichtung erhalten. Diese Lösung führt zu der Nutzung von bezüglich der Logik komplexeren Schieberegistern, also eigentlich weniger

leistungsfähigen, was die Transfergeschwindigkeit betrifft.

Da die Transfergeschwindigkeit sehr hoch sein muß, werden die Schieberegister  $RDM_j$  und  $RDP_j$ , ihre  
 5 zugeordneten Steuerlogiken  $TFR$  und  $TFR'$  und Freigabelogiken  $LV_1$  und  $LV_2$  aus einer schnellen Technologie ausgewählt (ECL, ASGA) und durch einen Taktgeber mit einer Frequenz  $F$  von wenigstens 100 MHz synchronisiert.

10 Eine andere, in Abbildung 21 dargestellte Lösung mit multiplexierten Registern ermöglicht, wie man später verstehen wird, eine erhebliche Verringerung der notwendigen leistungsfähigen, also teuren, Logik.

Das Mehrprozessorsystem der Abbildung 1 war  
 15 gleichzeitig mit einem gemeinsamen Bus für die Übertragung von Blockadressen und mit Serienverbindungen für den Datentransfer ausgestattet. Abbildung 6 stellt als Variante ein Mehrprozessorsystem mit demselben allgemeinen Prinzip dar, bei dem jedoch Daten und Adressen über die  
 20 Serienverbindungen ohne gemeinsamen Bus übertragen werden.

Dieses System umfaßt außer den Speicherregistern  $RDM_j$  zusätzliche Schieberegister  $RDC_j$ , die geeignet sind, die angeforderten Adressenblöcke zu speichern und die durch eine Logik des Typs  $TFR_j$  gesteuert  
 25 werden. Außerdem ist ein Arbitrer für die Zugriffsverwaltung ABM mit dem Hauptspeicher RAM und mit den Zusatzregistern  $RDC_j$  durch ihren parallelen Ausgang verbunden. Jede Logik  $TFR_j$  ist mit diesem herkömmlich strukturierten Arbitrer ABM verbunden. Der  
 30 Verwaltungsprozessor  $PG_j$  von jedem Cache-Speicher  $MC_j$  ist deshalb mit einem Teil des parallelen Eingangs des Prozessorregisters  $RDP_j$  verbunden, um auf diesen für das Schreiben Zugriff nehmen zu können.

Um einen Block  $b_i$  im Hauptspeicher RAM zu  
 35 lesen, stellt der Verwaltungsprozessor  $PG_j$  die Adresse des angeforderten Blocks und die Art der Anforderung (durch ein Präfix-Bit: 1' = Lesen, 0 = Schreiben) in den ihm zugänglichen Teil des Prozessorregisters  $RDP_j$ , wodurch der

Transfer dieser Information initialisiert wird. Die Transferlogik  $TFR_j$  entdeckt das Transferende auf dem Zusatzregister  $RDC_j$  und aktiviert eine Operations-Anforderung zum Arbiter  $ABM$ ; dieser hat die Aufgabe der Parallel-Seriell-Umsetzung der Lese-Anforderung des Blocks und deren Verarbeitung im Hauptspeicher  $RAM$ , indem er die Adresse des im Zusatzregister  $RDC_j$  angeforderten Blocks liest, das der vom Arbiter  $ABM$  gewählten Transferlogik entspricht und indem er dann den Block im Hauptspeicher  $RAM$  liest, der danach in das Speicherregister  $RDM_j$  geladen und wie zuvor übertragen wird.

Um einen Block in den Hauptspeicher  $RAM$  zu schreiben, führt der Verwaltungsprozessor  $PG_j$  nacheinander durch das Prozessorregister  $RDP_j$  die Übertragung der Adresse und dann des zu schreibenden Blocks aus. Das Zusatzregister  $RDC_j$  empfängt somit zuerst die Adresse und die Art der Anforderung.

Die Transferlogik  $TFR_j$  analysiert diese Anforderung und gibt den Empfang des Blocks im Speicherregister  $RDM_j$  aufgrund der Art der Anforderung (Schreiben) frei. Die Transferlogik  $TFR_j$  wird vom Ende des Transfers des Blocks  $bi$  informiert und überträgt dann an den Arbiter  $ABM$  seine Dienstabfrage. Diese Anforderung wird vom besagten Arbiter, der das Schreiben des Blocks  $bi$  in den Speicher aktiviert, verarbeitet, sobald er an der Reihe ist.

Im übrigen umfaßt das in Abbildung 7 dargestellte Mehrprozessorsystem Mittel für die Verwaltung der geteilten Daten, die es ermöglichen, statisch das herkömmliche Problem der Aufrechterhaltung der Kohärenz der geteilten Daten zu lösen. Dieses System umfaßt die Systemhilfsmittel der Abbildung 1 (dieselben Bezeichnungen) mit den folgenden Logiken und Zusatz-Hilfsmitteln:

Ein spezieller Bus für die parallele Wortübertragung  $BUSD$  verbindet die Prozessoren  $CPU_j$  und den Hauptspeicher  $RAM$ . Eine Partitionslogik  $LP_j$  ist jedem Prozessor  $CPU_j$  zugeordnet; jede Logik  $LP_j$  besteht

herkömmlicherweise aus mehreren Register-Komparator-Paaren, die parallel an den Adreßbus  $adr$  des Prozessors  $CPU_j$  angeschlossen sind, um eine Aufteilung des Speicherbereichs des Hauptspeichers  $RAM$  in Zonen für nicht  
 5 geteilte Daten und geteilte Daten durchzuführen; dazu gibt die besagte Logik  $LP_j$  ein Signal  $p$  aus (das die Art der Daten, geteilt oder nicht, angibt). Eine Decodierungslogik  $DEC$  ist dem Hauptspeicher  $RAM$  zugeordnet, der selbst so angeordnet ist, daß er von der besagten Logik  $DEC$  für das  
 10 Schreiben in Wörtern oder Blöcken gesteuert werden kann.

Die Decodierungslogik  $DEC$  ist in Abbildung 8 detailliert dargestellt und umfaßt einen Decodierer  $DECL$ , der auf seinem Dateneingang den Wortadresteil  $adrm$  der Adresse  $adr$  empfängt und der durch seinen Freigabeeingang  
 15 mit dem Ausgang einer Verknüpfung  $ET3$  verbunden ist, wobei jeder Ausgang  $i$  des besagten Decodierers mit einem Ausgangsfreigabe-"Puffer"  $BFS_i$  verbunden ist. Die Verknüpfung  $ET3$  empfängt auf ihren Eingängen das Signal  $p$  und das invertierte Signal  $\bar{r}$ . Ein Decodierer  $DECE$  ist auch  
 20 durch seinen Dateneingang mit dem Bus  $adrm$  und durch seinen Freigabeeingang an den Ausgang einer Verknüpfung  $ET4$  verbunden, seine Ausgänge sind mit mehreren Verknüpfungen  $OUI_i$  deren Anzahl genauso groß ist, wie die der Wörter in einem Block, verbunden. Die Verknüpfung  $ET4$   
 25 empfängt auf ihren Eingängen das Signal  $p$  und das invertierte Signal  $\bar{w}$ . Der Ausgang der Verknüpfung  $ET4$  ist ebenfalls mit mehreren Eingangsfreigabe-"Puffern"  $BFE_i$ ,  $BFE_i \dots$  verbunden. Der Hauptspeicher  $RAM$  kann für das Wort-Schreiben gesteuert werden. Jede so definierte Wort-  
 30 "scheibe" hat ihren Schreibsteuereingang  $w_i$ . Der Ausgang jeder Verknüpfung  $OUI_i$  ist mit dem Eingang  $w_i$  jeder Wort-"scheibe" des Hauptspeichers  $RAM$  verbunden.

Abbildung 8 stellt außerdem im Detail die Adressierung der Speicherregister  $RDM_j$  dar, die an erster  
 35 Stelle einen Decodierer  $DECEB$  umfaßt, der durch seinen Dateneingang mit dem gemeinsamen Bus  $BUSA$  verbunden ist, um die Nummer  $j$  des von der Abfrage der Zentraleinheit  $UC_j$  betroffenen Prozessors zu empfangen; dieser Decodierer

DECEB ist durch seinen Freigabeeingang mit dem Ausgang einer Verknüpfung ET5 und durch seine Ausgänge 1, 2 ... j mit Freigabe-"Puffern"  $BV_1, BV_j \dots$  verbunden. Die Verknüpfung ET5 empfängt auf ihren Eingängen die invertierten Signale  $p$  und  $\bar{w}$ . Gleichmaßen ist auch ein Decodierer DECLB durch seinen Dateneingang mit dem Feld j des gemeinsamen Busses BUSA und durch seinen Freigabeeingang mit dem Ausgang einer Verknüpfung ET6 verbunden; die Ausgänge 1, 2 ... j dieses Decodierers DECLB sind mit den Ladeeingängen  $ld_1$  und  $ld_j$  der Speicher-Schieberegister  $RDM_j$  verbunden. Die Verknüpfung ET6 empfängt auf ihren Eingängen die invertierten Signale  $p$  und  $\bar{r}$ .

Das System funktioniert wie folgt: bei jeder Speicherreferenz liefert der Prozessor CPU<sub>j</sub> eine Adresse auf seinen Adreßbus  $adr$  und die Abfrageart: Lesen  $\bar{r}$  oder Schreiben  $\bar{w}$ . Für das Lesen erwartet er Daten und für das Schreiben liefert er Daten. Die Adresse  $adr$  geht durch die Partitionslogik  $LP_j$ , die durch das Signal  $p$  angibt, ob die Adresse  $adr$  zu einem Bereich von nicht geteilten Daten ( $p = 0$ ) oder von geteilten Daten ( $p = 1$ ) gehört. Im ersten Fall wird die Anforderung zum Verwaltungsprozessor  $PG_j$  geleitet und gemäß dem in Abbildung 1 beschriebenen Betriebsmodus verarbeitet. Im zweiten Fall wird die Abfrage direkt zum gemeinsamen Bus BUSA geleitet; der Bus der Adresse  $adr$  enthält zusätzliche Adressendrähte für die Identifizierung des betreffenden Worts: die Adresse  $adr$  besteht zu einem Teil aus der Blockadresse  $adrb$  und zum anderen Teil aus der Wortadresse  $adrm$ . So empfängt der Hauptspeicher RAM, nach Erlaubnis des Busarbiters AB, entweder eine Anforderung für eine Blocktransaktion ( $p = 0$ ), und in diesem Fall ist nur der Blockteil  $adrb$  der Adresse  $adr$  von Bedeutung, oder eine Anforderung für eine Worttransaktion ( $p = 1$ ), und in diesem Fall ist die ganze Adresse  $adr$  (Block  $adrb$  und Wort  $adrm$ ) von Bedeutung.

Für das Lesen eines Blocks,  $p = 0$  und  $r = 0$ , gibt die Verknüpfung ET6 den Decodierer DECLB frei, der ein Ladesignal  $LD_j$  an das Schieberegister  $RDM_j$  ausgibt,

was das Laden des Blocks in diesen letzteren ermöglicht, nachdem der Block an der Adresse  $adrb$  durch das Lesesignal  $\bar{r}$  im Hauptspeicher RAM gelesen wurde.

Für das Schreiben eines Blocks,  $p = 0$  und  $w = 0$ , gibt die Verknüpfung ET5 den Decodierer DECEB frei, der ein Freigabesignal auf dem "Puffer"  $BV_j$  ausgibt; dadurch kann der Inhalt dieses Registers dem Hauptspeicher RAM vorgelegt und somit an der Adresse  $adrb$  geschrieben werden, wobei das Schreibsignal des Blocks vom Ausgang der Verknüpfung ET5 geliefert wird. Dieses Signal wird durch die Verknüpfungen  $OU1_i$  auf den Schreibeingängen  $w_1, w_i, \dots$  an die Wort-"scheiben" des Hauptspeichers RAM ausgesandt.

Für das Lesen eines Worts,  $p = 1$  und  $r = 0$ , gibt die Verknüpfung ET3 den Decodierer DECL frei, der ein Freigabesignal auf dem "Puffer"  $BFS_i$  ausgibt, wodurch das angeforderte Wort (der Adresse  $adrm$  im Block  $adrb$ ), das durch das Signal  $\bar{r}$  gelesen wird, zu dem speziellen Übertragungsbus BUSD geleitet werden kann. Dieses Wort wird direkt von dem Prozessor  $CPU_j$  auf seinen Dateneingang  $data$  aufgenommen.

Für das Schreiben eines Worts,  $p = 1$  und  $w = 0$ , gibt die Verknüpfung ET4 den Decodierer DECE frei, der auf seinem Ausgang  $i$  ein Signal liefert, das durch die Verknüpfung  $OU1_i$  zum Schreibeingang  $w_i$  der Wort"scheibe" des betreffenden Hauptspeichers RAM geleitet wird; dieses, am Eingang  $w_i$  vorhandene Signal ermöglicht das Schreiben des von dem Prozessor  $CPU_j$  auf den Datenbus BUSD gelieferten Worts nur in diese Wort-"scheibe". Der Inhalt dieses Busses wird parallel auf allen Wort"scheiben" des Hauptspeichers RAM vorgelegt, und zwar dank der Aktivierung der "Puffer"  $BFE_i$  durch das aus der Verknüpfung ET4 kommende Signal.

Eine wesentliche Eigenschaft der erfindungsgemäßen Architektur ist die minimale Abfragelast auf dem gemeinsamen Bus BUSA. In der in Abbildung 7 schematisierten Architektur wird der gemeinsame Bus BUSA von Blockadressen und Wortadressen beansprucht. Die

Frequenz der Anforderungen nach Wortadressen ist abhängig von der Rate der geteilten Daten und kann zu einer Sättigung des gemeinsamen Busses BUSA führen.

Abbildung 9 stellt in einer Variante eine Lösung zur Verringerung dieser Last dar. Das angestrebte System umfaßt außer den Hilfsmitteln der Abbildung 7, einen Bus BUSAM für die Wortadressen, einen Arbiter AB' zur Lösung der Zugriffskonflikte auf den Bus BUSAM, einen Arbiter ABM zur Lösung der Zugriffskonflikte, die von den Bussen BUSA und BUSAM kommen, und ist mit einem Multiplexer MUX verbunden, der selbst durch seine Eingänge mit den beiden Bussen BUSA und BUSAM verbunden ist.

Dieses System funktioniert wie folgt: wie zuvor liefert die Partitionslogik LP<sub>j</sub> das Signal p zur Identifizierung der Art der manipulierten Daten.

Betrifft die Anforderung nicht geteilte Daten ( $p = 0$ ), dann bewirkt jedes Fehlen von Informationen eine Speicherabfrage von dem Blocktyp, die durch den gemeinsamen Bus BUSA übertragen wird.

Betrifft die Anforderung geteilte Daten ( $p = 1$ ), wird die Abfrage zum gemeinsamen Bus BUSAM geleitet. So kann der Hauptspeicher RAM auf den beiden Bussen BUSA und BUSAM gleichzeitige Anforderungen empfangen, die geschichtet sein müssen. Der Arbiter ABM erteilt einer der beiden Abfragen auf herkömmliche Art und Weise den Zugriff auf den Hauptspeicher RAM und rekonstituiert das Signal p anhand der Herkunft der Anforderung ( $p = 0$  für BUSA,  $p = 1$  für BUSAM). Das Signal p steuert dann einerseits den Multiplexer MUX, der die Signale des von der Abfrage betroffenen Busses durchläßt, und andererseits die Decodierungslogik DEC: man befindet sich in derselben Situation, wie der des vorhergegangenen Systems.

Es ist festzustellen, daß sich die Last vom gemeinsamen Bus zum Hauptspeicher RAM verlagert hat, da die Rate der Abfragen bei diesem letzteren dieselbe bleibt und seine Zyklusdauer von der gleichen Größenordnung ist oder sogar noch über der des Buszyklus liegt.

Diese Lösung ist also nur interessant, wenn der Hauptspeicher RAM aus unabhängigen Hauptspeicherbänken besteht, die gemäß der später gegebenen diesbezüglichen Beschreibung der Abbildung 20 angeordnet sind: mehrere  
 5 Transaktionen können in diesem Fall gleichzeitig stattfinden, wenn sie verschiedene Speicherbänke betreffen.

Abbildung 10 stellt ein synoptisches Schema einer erfindungsgemäßen Architektur-Durchführungsart dar,  
 10 bei der das Problem der geteilten Daten dynamisch gelöst wird. Dazu besitzt die dieser Durchführungsart entsprechende Vorrichtung einen Busspion-Prozessor  $PE_j$ , der an einen Verwaltungsprozessor des parallelen Drahts  $PGP_j$  gekoppelt ist. Ein Verwaltungsprozessor des  
 15 Seriendrahts  $PGS_j$  ist mit dem Spion-Prozessor  $PE_j$  durch eine Warteschlange  $FIFO_j$  verbunden. Ein Prozessor für die Verwaltung von Abfragen der Zentraleinheit  $PGU_j$  ist einerseits mit dem Verarbeitungs-Prozessor  $CPU_j$  und andererseits mit den Prozessoren für die Verwaltung des  
 20 parallelen Drahts  $PGP_j$  und die Verwaltung des Seriendrahts  $PGS_j$  verbunden. Die dem Verwaltungsprozessor  $PG_j$  von jedem Cache-Speicher entsprechende Logik ist in dieser Durchführungsart in die verschiedenen oben dargestellten Prozessoren aufgeteilt. Der Zugriff auf den Cache-Speicher  
 25  $MC_j$  und dessen Verzeichnis  $RG_j$  wird durch einen Prozessor für die Verwaltung des Verzeichnisses und des Caches-Speichers  $PGR_j$  geregelt.

Schließlich ist ein Prozessor für die Verwaltung des Hauptspeichers RAM  $PGM$  an den Bus  $BUSA$  und  
 30 an den Hauptspeicher RAM und an dessen Schieberegister  $RDM_j$  angeschlossen.

Das Ganze funktioniert folgendermaßen:

Jede Transaktion auf dem gemeinsamen Bus  $BUSA$  entspricht einer Lese- oder Schreib-Anforderung eines  
 35 Blocks  $bi$ . Die Busspion-Prozessoren  $PE_j$  werden durch jede Lese-Anforderung eines Datenblocks aktiviert. Diese Operation, die innerhalb desselben Zyklus von allen Spionprozessoren durchgeführt wird, ermöglicht es, die

Einzigartigkeit der Werte der geteilten Daten zu garantieren. Der Spionprozessor  $PE_j$  verfügt über einen Zugang zum Verzeichnis  $RG_j$ . Die Anwendungsfunktion, die für die Verwaltung des Cache-Speichers  $MC_j$  verwendet wird, ist in der beschriebenen Durchführungsart vom Typ einer direkten Anwendung. Jedes Element des Verzeichnisses ist ein Blockstichwort, das ein Feld "tag" (Blockadresse) und herkömmliche Zustandsbits des Blocks enthält: ein Freigabebit  $v$  und ein Modifikationsbit  $m$  und zwei Zusatzbits  $a$ , um anzuzeigen, daß der Block dem Cache-Speicher bekannt ist, jedoch noch im Zuge des Transfers auf der Serienverbindung, und  $f$ , um anzuzeigen, daß sich der Block in der Warteschlange FIFO befindet, um so zu verhindern, daß er mehrmals dort plazierte wird.

Der Speicher-Verwaltungsprozessor PGM verfügt einerseits über eine assoziativ zugängliche Warteschlange AFIFO für Adressen von Blöcken  $b_i$  und für Prozessoradressen, andererseits über ein Verzeichnis der Blockzustände, die aus 2 Bits pro Block  $ro$  und  $rw$  bestehen und die die folgenden möglichen Blockzustände angeben:

- .  $ro = rw = 0$ : Block noch nicht ausgesandt,
- .  $ro = 1; rw = 0$ : Block bereits für das Lesen ausgesandt: eine oder mehrere Kopien dieses Blocks befinden sich in den Cache-Speichern,
- .  $ro = 0; rw = 1$ : Block für das Schreiben ausgesandt: die aktualisierte Kopie dieses Blocks befindet sich in einem Cache-Speicher.

Die Entwicklung der Blockzustandsbits ist die folgende und je nach Art der Anforderung des Verarbeitungs-Prozessors  $CPU_j$  unterschiedlich:

- fordert der Prozessor  $CPU_j$  das Lesen nicht geteilter Daten an (Bereich für Programm oder für die explizit nicht geteilten Daten): der Block wird im Hauptspeicher als bereits für das Lesen ausgesandt beim Transfer des besagten Blocks vom Hauptspeicher RAM zum Register  $RDM_j$  ( $ro = 1; rw = 0$ ) und als nicht modifiziert ( $m = 0$ ) im Cache-Speicher innerhalb desselben Update-Zyklus des Verzeichnisses  $RG_j$  des Cache-Speichers

(gültiger Block) gekennzeichnet. Die Spione haben nicht auf die Abfrage auf dem gemeinsamen Bus reagiert (da die Anforderung mit dem Hinweis "nicht geteilte Daten" gemacht wurde),

5                   - fordert der Prozessor  $CPU_j$  das Lesen der Daten (apriori geteilt) an, dann ist der Bus BUSA während der Dauer der Übergabe der Adresseninformationen und der Art der Abfrage, der Dauer ihrer Verarbeitung durch den Prozessor PGM und die Spione des gemeinsamen Busses  $PE_j$   
10 belegt. Im Hauptspeicher RAM kann dieser Block:

1. Noch nicht ausgesandt sein:  $ro = rw = 0$ . Er wird dann an die Zentraleinheit  $UC_j$  übertragen und nimmt den nicht modifizierten Zustand an,

2. Bereits für das Lesen ausgesandt sein:  $ro =$   
15 1;  $rw = 0$ . Er wird dann an die Zentraleinheit  $UC_j$  übertragen. Sein Zustand ändert sich nicht,

3. Bereits für das Schreiben ausgesandt sein:  
 $ro = 0$ ;  $rw = 1$ . Die aktualisierte Kopie dieses Blocks befindet sich in einem Cache-Speicher  $MC_i$ . Der diesem  
20 Cache-Speicher zugeordnete Spionprozessor  $PE_i$  hat die Anforderung der Zentraleinheit  $UC_j$  während der Übergabe der Adresse auf den gemeinsamen Bus registriert und den Transfer des Blocks in den Hauptspeicher RAM sobald wie möglich auf seiner Serienverbindung  $LS_i$  vorgenommen. Bis  
25 zum effektiven Transfer stellt der Speicher-Verwaltungsprozessor PGM die Anforderung in die assoziative Warteschlange, die eine Anzahl von Stellen enthält die gleich der Anzahl der Prozessoren ist.

Bei der Lese-Anforderung auf dem gemeinsamen  
30 Bus BUSA haben alle Spionprozessoren  $PE_i$  durch Konsultation des ihrem Cache-Speicher  $MC_i$  zugeordneten Verzeichnisses  $RG_i$  reagiert. Der gemeinsame Bus BUSA wird nur freigegeben, wenn alle Spionprozessoren  $PE_i$  ihren Zugriff auf das Verzeichnis  $RG_i$  genommen haben,  
35 was denselben Blockzustand im ganzen System garantiert. Der Prozessor, der die aktualisierte Kopie in seinem Cache-Speicher hat, führt, sobald seine Serienverbindung frei ist, den Transfer dieses Blocks in das Register  $RDM_i$

aus und führt eine Schreib-Anforderung des Blocks auf dem gemeinsamen Bus durch, wodurch die in der assoziativen Warteschlange AFIFO wartende Anforderung freigemacht wird und das Update der Blockzustandsbits erfolgt.

5 Für das Update des Blocks wird also lediglich das Schreiben in den Hauptspeicher RAM ohne Aktivierung der Spione benötigt.

- Fordert der Prozessor CPU<sub>j</sub> das Schreiben von Daten in einen in seinem Cache-Speicher MC<sub>j</sub> vorhandenen  
10 Block mit dem nicht veränderten Zustand an, dann muß auf dem gemeinsamen Bus BUSA eine Informations-Schreib-Anforderung gesandt werden, da es möglich ist, daß andere Cache-Speicher MC<sub>i</sub> diesen Block in demselben Zustand besitzen. Diese anderen Speicher müssen von der  
15 Zustandsänderung informiert werden. Zu diesem Zweck konsultieren alle Spionprozessoren PE<sub>i</sub> (durch den auf dem gemeinsamen Bus BUSA ausgesandten Informations-Schreibvorgang aktiviert) ihr Verwaltungsverzeichnis RG<sub>i</sub> und machen diesen Block ungültig, während der  
20 Hauptspeicher und der parallele Verwaltungsprozessor PGP<sub>j</sub> zu derselben Zeit die Zustandsänderung dieses Blocks im Verwaltungsverzeichnis RG<sub>j</sub> registrieren. Das Freimachen des gemeinsamen Busses BUSA durch alle Spionprozessoren und den Hauptspeicher RAM ermöglicht dem Prozessor CPU<sub>j</sub>  
25 die Durchführung des Schreibvorgangs in seinen Cache-Speicher MC<sub>j</sub>, da das Update des Zustandsbits des Verwaltungsverzeichnisses RG<sub>j</sub> erfolgt ist.

Wartet eine Zentraleinheit auf einen Zugriff auf den Bus BUSA bezüglich derselben Abfrage auf demselben  
30 Block, so wandelt sich ihre Anforderung in einfaches Schreiben und folgt dann dem Anforderungsprotokoll für das Schreiben eines Blocks.

- Fordert der Prozessor CPU<sub>j</sub> das Schreiben von Daten in einen im Cache-Speicher MC<sub>j</sub> nicht vorhandenen  
35 Block an, dann wird dieser Block im Hauptspeicher RAM gelesen und in den Cache-Speicher MC<sub>j</sub> geleitet, damit das Schreiben effektiv wird.

Im Hauptspeicher RAM kann dieser Block:

1. Noch nicht ausgesandt sein:  $ro = rw = 0$ .  
 Der Block wird dann auf der Serienverbindung  $LS_j$  in den  
 Cache-Speicher  $MC_j$  gesandt. Er nimmt im Hauptspeicher die  
 Zustände  $ro = 0$ ;  $rw = 1$  und im Cache-Speicher den  
 5 "modifiziert"-Zustand ( $m = 1$ ) an,

2. Bereits zum Lesen ausgesandt sein:  $ro = 1$ ;  
 $rw = 0$ . Der Block wird auf der Serienverbindung  $LS_j$  in den  
 Cache-Speicher  $MC_j$  gesandt. Er nimmt im Hauptspeicher die  
 Zustände  $ro = 0$ ;  $rw = 1$  und im Cache-Speicher den  
 10 "modifiziert"-Zustand ( $m = 1$ ) an. Bei der Anforderung auf  
 dem gemeinsamen Bus  $BUSA$ , haben die Spionprozessoren  $PE_i$   
 die Abfrage registriert und diese Blocknummer in ihrem  
 Cache-Speicher  $MC_i$  ungültig gemacht,

3. Bereits zum Schreiben ausgesandt sein:  $ro =$   
 15  $0$ ;  $rw = 1$ . Die Anforderung wird in die assoziative  
 Warteschlange AFIFO gestellt und der gemeinsame Bus  $BUSA$   
 freigemacht.

Der Spionprozessor  $PE_i$  des Cache-Speichers  
 $MC_i$ , Besitzer der aktualisierten Kopie, aktiviert sobald  
 20 wie möglich den Transfer des angeforderten Blocks aus  
 seinem Cache-Speicher  $MC_i$  in den Hauptspeicher RAM. Dieser  
 Block wird dann im Cache-Speicher  $MC_i$  ungültig gemacht.

Die Zentraleinheit  $UC_j$  führt eine Schreib-  
 Anforderung für ein Block-Update in den beiden folgenden  
 25 Fällen durch:

a) der Cache-Speicher ist gesättigt und die  
 vollständige Entladung eines Blocks benötigt das Update  
 dieses Blocks im Hauptspeicher,

b) eine Zentraleinheit  $UC_i$  erwartet einen  
 30 Block, dessen einzige aktualisierte Kopie sich im Cache-  
 Speicher  $MC_j$  befindet. Der Spionprozessor  $PE_j$  registriert  
 die Anforderung und führt sobald wie möglich die  
 vollständige Entladung dieses Blocks durch.

Seitens des Hauptspeichers RAM bewirkt jede  
 35 Update-Schreib-Anforderung eine Konsultation der  
 assoziativen Warteschlange AFIFO und, bei Entdecken einer  
 auf diesen Block wartenden Zentraleinheit  $UC_i$ , das Laden  
 dieses Blocks in das Schieberegister  $RDM_i$  und das Update

der diesem Block entsprechenden Zustandsbits. Diese Art der Schreib-Anforderung beansprucht also nicht die Spionprozessoren.

Der Verzeichnis-Verwaltungsprozessor  $PGR_j$ , der bei dieser Durchführungsart hinzugefügt wird, ermöglicht den Ablauf des oben angesprochenen Algorithmus, indem er die Zugriffe auf das Verzeichnis des Cache-Speichers  $MC_j$  koordiniert und die Abfrage von drei funktionellen asynchronen Einheiten empfängt:

1. Vom Verarbeitungs-Prozessor  $CPU_j$ , um die Instruktionen des gerade ausgeführten Programms zu lesen und die von diesem Programm manipulierten Daten zu lesen oder zu schreiben,

2. Vom gemeinsamen Bus  $BUSA$ , um die Kohärenz der Daten im Cache-Speicher  $MC_j$  aufrechtzuerhalten,

3. Von der Serienverbindung  $LS_j$ , um einen Informationsblock aus/in den Hauptspeicher  $RAM$  zu laden/zu entladen.

Jede dieser Abfragen nimmt Zugriff auf das Verzeichnis  $RG_j$  des Cache-Speichers. Die Parallel-Seriell-Umsetzung dieser Zugriffe auf das genannte Verzeichnis ermöglicht den richtigen Ablauf des oben angesprochenen Algorithmus der Informationskohärenz in den Cache-Speichern. So erhält man eine starke Kopplung der Abfrage beim Verzeichnis  $RG_j$ , jedoch ist die Synchronisation, die bei der Verarbeitung dieser Abfrage bestehen muß, schwach genug, um einen asynchronen Betrieb der Verarbeitungslogik dieser Abfragen ins Auge zu fassen, was zu der folgenden funktionellen Aufteilung führt:

Die Schnittstelle eines jeden Prozessors  $CPU_j$  und seiner Hilfsprozessoren ( $PGS_j$ ,  $PGU_j$ ) mit dem gemeinsamen Bus  $BUSA$  besteht aus zwei Teilen, die nur funktionieren, wenn sie sich gegenseitig ausschließen: der Verwaltungsprozessor des parallelen Drahts  $PGP_j$  hat die Aufgabe, den gemeinsamen Bus  $BUSA$  auf Anforderung des Abfrage-Verwaltungsprozessors  $PGU_j$  oder des Verwaltungsprozessors des Seriendrahts  $PGS_j$  zu fordern und

den gemeinsamen Bus BUSA für den Schreibvorgang zu steuern. Der Busspion-Prozessor  $PE_j$  führt die Spionageaufgabe aus, was darauf hinausläuft, den gemeinsamen Bus BUSA für den Lesevorgang zu steuern. Er  
 5 nimmt oft Zugriff auf das Verzeichnis  $RG_j$  des Cache-Speichers  $MC_j$ .

Der Verwaltungsprozessor des Seriendrahts  $PGS_j$  verwaltet die Schnittstelle mit der Serienverbindung  $LS_j$ . Er führt das Laden und Entladen von Informationsblöcken bei  
 10 auf Anforderung des Abfrage-Verwaltungsprozessors  $PGU_j$  und des Busspion-Prozessors  $PE_j$  aus. Er nimmt selten Zugriff auf den Cache-Speicher  $MC_j$  und auf das entsprechenden Verwaltungsverzeichnis  $RG_j$ .

Der Abfrage-Verwaltungsprozessor  $PGU_j$  sorgt  
 15 für den Fortlauf der Abfrage aus dem Prozessor  $CPU_j$ . Er nimmt sehr oft Zugriff auf den Cache-Speicher  $MC_j$  und auf das Verwaltungsverzeichnis  $RG_j$ . Diese Schnittstelle beinhaltet die eventuelle Logik für die "MMU" ("Memory Management Unit"), die gewöhnlich dem Verarbeitungs-  
 20 Prozessor  $CPU_j$  zugeordnet ist.

Der Verwaltungsprozessor  $PGR_j$  des Verwaltungsverzeichnisses  $RG_j$  ist der mit der Zuteilung des Zugriffs auf den Cache-Speicher  $MC_j$  beauftragte Arbiter.

25 Die Abbildungen 11, 12, 13, 14, 15, 16 und 17 stellen als Beispiele Durchführungsarten der verschiedenen funktionellen Einheiten der Vorrichtung von Abbildung 10 dar. Die Bezeichnungen der Signale oder Ein- und Ausgänge der Einheiten sind wie gewohnt gewählt worden. Signale mit  
 30 derselben Funktionalität, die in jeder funktionellen Einheit von einem Basissignal aus erzeugt werden, werden mit derselben Referenz bezeichnet, zum Beispiel: dnp = nicht geteilte Daten, dl = Lese-Anforderung, maj = Update, de = Schreib-Anforderung, ei = Informations-  
 35 Schreibvorgang. Die Vorrichtung besitzt mehrere Prozessoren und der untere Index  $j$  hat bisher einen gängigen Prozessor und dessen Peripheriegeräte bezeichnet; um die Beschreibung gefälliger zu machen, wurde dieser

untere Index auf diesen Abbildungen weggelassen und es ist selbstverständlich, daß die folgende Beschreibung alle funktionellen Einheiten betrifft, die mit jedem Verarbeitungs-Prozessor verbunden sind. Im übrigen  
 5 definieren die mit  $x_{YZ}$  bezeichneten Signale den Namen und die Herkunft des Signals für den Fall wo  $YZ = RG, MC, UC$  ist, und in den anderen Fällen den Sender und den Empfänger des Signals, wobei  $Y$  und  $Z$ :  $U = PGU, R = PGR, P = PGP$  oder  $PE, S = PGS$  sind.

10 Der in Abbildung 11 dargestellte Cache-Speicher MC hat zum Beispiel eine Kapazität von 16 KByte. Er ist in 16 schnelle Schreib-Lese-Speicher-Module von je 1 KByte angeordnet,  $MC_0, \dots, MC_{15}$ , wobei jedes auf einer Breite von 4 Byte zugänglich ist: der Adreßbus des Cache-  
 15 Speichers MC (mit  $adr_{MC}$  bezeichnet) umfaßt einen Blockadresteil  $adr_{bloc}$  und einen Wortadresteil in dem Block  $adr_{mot}$ . Der Adreßbus  $adr_{MC}$  besteht aus 14 Drähten, die es ermöglichen, die 16 KByte des Cache-Speichers MC zu adressieren. Der Teil  $adr_{bloc}$  umfaßt 8 Drähte, die eine  
 20 Adressierung der 256 Blockstellen des Cache-Speichers MC ermöglichen, und der Teil  $adr_{mot}$  6 Drähte, die es ermöglichen, ein Wort in den in dem Beispiel 64-Byte-Block zu adressieren.

Der Adreßteil  $adr_{bloc}$  ist mit dem  
 25 Adreßeingang jedes Speichermoduls  $MC_0 \dots MC_{15}$  verbunden. Der Wortadresteil  $adr_{mot}$  ist mit dem Eingang von zwei Decodierern  $DECO$  und  $DEC1$  verbunden (nur die 4 höchstwertigen Bits des Adreßbusses  $adr_{mot}$  werden genutzt: die Adresse ist eine Byte-Adresse und der Cache-  
 30 Speicher hat als Zugriffseinheit ein Wort von 4 Byte). Das Lesesignal  $\overline{r}_{MC}$  wird an jeden Leseingang der Speichermodule  $MC_0 \dots MC_{15}$  und an einen der Eingänge einer Verknüpfung  $OU1$  ausgegeben. Der andere Eingang dieser Verknüpfung  $OU1$  empfängt das invertierte Signal  
 35  $bloc$ . Das Schreibsignal  $\overline{w}_{MC}$  wird an einen der beiden Eingänge von Verknüpfungen  $OU2$  und  $OU3$  ausgegeben. Die Verknüpfung  $OU2$  empfängt auf ihrem anderen Eingang das invertierte Signal  $bloc$ . Die Verknüpfung  $OU3$  empfängt auf

ihrem anderen Eingang das Signal  $\overline{\text{bloc}}$ . Der Ausgang der Verknüpfung OU1 ist mit dem Freigabeeingang  $\overline{\text{en1}}$  des Decodierers DEC1 verbunden, und der Ausgang des Rangs i dieses Decodierers DEC1 aktiviert einen Freigabe-"puffer" BVL vom Rang i. Der Ausgang der Verknüpfung OU2 ist mit dem Freigabeeingang  $\overline{\text{en0}}$  des Decodierers DEC0 und mit Freigabe-"puffern" BVE verbunden. Der Ausgang der Verknüpfung OU3 ist mit Verknüpfungen ET1<sub>0</sub> ... ET1<sub>15</sub> verbunden, die auf ihrem anderen Eingang den entsprechenden Ausgang, der dem Rang des Decodierers DEC0 entspricht, empfangen. Der Ausgang i jeder Verknüpfung ET1<sub>0</sub> ... ET1<sub>15</sub> ist mit dem Schreibeingang  $\overline{\text{w}}_0$  ...  $\overline{\text{w}}_{15}$  eines jeden Speichermoduls MC<sub>0</sub> ... MC<sub>15</sub> verbunden. Ein Datenbus verbindet jedes Speichermodul MC<sub>0</sub> ... MC<sub>15</sub> mit einem der Freigabepuffer BVL und mit einem der Freigabepuffer BVE. Der Ausgang der Puffer BVL und der Eingang der Puffer BVE empfangen parallel einen Datenbus  $\text{datamot\_MC}$  (der mit dem Abfrage-Verwaltungsprozessor PGU verbunden ist).

Der im obigen Beispiel beschriebene Cache-Speicher funktioniert wie folgt:

#### Fall 1

Die Anforderung kommt vom Verwaltungsprozessor des Seriendrahts PGS. Dieser Fall wird durch Vorhandensein eines logischen Null-Zustands auf dem Signal  $\text{bloc}$  signalisiert.

Beim Lesen des Cache-Speichers legt der Verwaltungsprozessor des Seriendrahts PGS dem Adreßbus  $\text{adr\_MC}$  die Adresse der zu lesenden Blockstelle vor (in diesem Fall wird nur der Teil  $\text{adr\_bloc}$  des Busses  $\text{adr\_MC}$  benutzt) und aktiviert das Lesesignal  $\overline{\text{r\_MC}}$ . Am Ende der Zugriffszeiten ist der Block auf dem Bus  $\text{databloc\_MC}$  verfügbar.

Beim Schreiben in den Cache-Speicher legt der Verwaltungsprozessor des Seriendrahts dem Adreßbus  $\text{adr\_MC}$  die zu schreibende Adresse der Blockstelle und dem Datenbus  $\text{databloc\_MC}$  die darin einzuschreibenden Daten vor und aktiviert die Leitung  $\overline{\text{w\_MC}}$ . Der Null-Zustand des Signals  $\text{bloc}$  leitet das Signal  $\overline{\text{w\_MC}}$  über die Verknüpfungen

OU3 und ET1<sub>1</sub> zu den Schreibsteuer-Eingängen der Module des Cache-Speichers MC<sub>0</sub> ... MC<sub>15</sub>. Die auf dem Datenbus databloc\_MC vorhandene Information wird am Ende der Schreibzeit in den Cache-Speicher eingeschrieben.

5                    Fall 2

Die Anforderung kommt vom Abfrage-Verwaltungsprozessor PGU des Verarbeitungs-Prozessors CPU. Dieser Fall wird durch Vorhandensein eines Logikzustands Eins auf dem Signal bloc signalisiert.

10                    Beim Lesen des Cache-Speichers legt der Verwaltungsprozessor PGU dem Bus adr\_MC die Adresse des angeforderten Wortes vor und aktiviert das Lesesignal  $\overline{r\_MC}$ . Der dem Teil adr\_bloc entsprechende Block wird im Cache-Speicher gelesen, und das angeforderte Wort wird  
15 über einen der Freigabepuffer BVL zum Datenbus datamot\_MC geleitet. Der betreffende Freigabepuffer BVL wird durch den Decodiererausgang DEC1 aktiviert, der der angeforderten Wortadresse adr\_mot entspricht.

Beim Schreiben in den Cache-Speicher legt der  
20 Verwaltungsprozessor PGU dem Bus adr\_MC die Adresse des zu schreibenden Wortes vor und dem Datenbus datamot\_MC die zu schreibenden Daten und aktiviert das Schreibsignal  $\overline{w\_MC}$ . Die auf dem Bus datamot\_MC vorhandenen Daten werden über den freigegebenen Puffer BVE durch das Schreibsignal auf  
25 jedes Modul des Cache-Speichers ausgesandt. Das Schreibsignal  $\overline{w\_MC}$  wird dann dem einzigen betroffenen Speichermodul vorgelegt. Es wird an den Ausgang des Decodierers DECO, der der betreffenden Adresse adr\_mot entspricht, ausgegeben.

30                    In der oben beschriebenen Durchführungsart sind die Zugriffsprobleme in Byte und Doppel-Byte, und des Zugriffs in Doppel-Byte und in einem sich auf zwei Speichermodule erstreckenden Wort auf dieselbe Art und Weise gelöst, wie in den traditionellen  
35 Datenverarbeitungssystemen und sind hier nicht beschrieben.

Die Abbildungen 12a, 12b, 12c und 12d stellen als Beispiel die Eigenschaften eines

Verwaltungsverzeichnisses des Cache-Speichers RG und eines zugeordneten Verwaltungsprozessors PGR dar. Abbildung 12a illustriert die logische Struktur der Adresse `adr_RG` in der Annahme eines mit 32 Bit adressierten Bereichs und mit den zuvor beschriebenen Eigenschaften des Cache-Speichers. Das Feld - tag -, Bestandteil des Adreßblocks, ist auf 18 Bit codiert. Das Feld - Rahmen- ist auf 8 Bit codiert und ermöglicht die Adressierung der 256 Blockstellen des Cache-Speichers MC. Die letzten 6 Bit definieren die Wortadresse in dem Block in Byte-Einheiten.

Abbildung 12b stellt die Struktur des Verwaltungsverzeichnisses des Cache-Speichers RG dar, der ein einfacher schneller Schreib-Lese-Speicher mit einer Kapazität von 256 Wörtern von je 22 Bit ist. Jedes Wort der Adresse `i` enthält das Stichwort des an der Stelle `i` des Cache-Speichers eingeschriebenen Blocks.

Abbildung 12c schematisiert die Struktur des Stichworts, das folgendes enthält:

- ein Feld `tag` von 18 Bit, das die Blockadresse in der gängigen Blockstelle oder dem gängigen Blockrahmen definiert,
- das Freigabebit `v`,
- das Modifikationsbit `m`,
- das Wartebit des Transferendes `a`,
- das Wartebit für die vollständige Entladung.

Abbildung 12d liefert die Struktur des Prozessors PGR, der nichts anderes ist, als ein herkömmlicher Arbiter mit feststehender Priorität.

Dieser Arbiter umfaßt ein LATCH-Register, von dem drei Eingänge das Signal `rqst_UR`, `rqst_PR` beziehungsweise `rqst_SR` empfangen, das ebenfalls zur Verknüpfung ET2, ET3 beziehungsweise ET4 ausgegeben wird. Die entsprechenden Ausgänge des LATCH-Registers sind mit den Eingängen eines Prioritätenkodierers PRI verbunden, dessen Ausgänge mit den Eingängen eines Decodierers DEC PRI verbunden sind. Die Ausgänge des LATCH-Registers, deren Rang denen der Ausgänge entsprechen, sind mit den Signalen `grnt_UR`, `grnt_PR` und `grnt_SR` verbunden sowie, invertiert,

mit den Eingängen der Verknüpfung ET2, ET3 beziehungsweise ET4. Die Ausgänge der Verknüpfungen ET2, ET3 und ET4 sind mit den Eingängen der Verknüpfung NOU1 verbunden. Der Ausgang der Verknüpfung NOU1 ist mit einer Kippschaltung

5 B1 verbunden, die auf ihrem Eingang D den Ausgang  $\overline{e0}$  des Prioritätencodierers PRI empfängt. Die ganze Vorrichtung wird durch einen allgemeinen Taktgeber synchronisiert, der ein Signal h an einen der Eingänge clk einer Verknüpfung ET5 ausgibt und, invertiert, an den Taktgebereingang der

10 Kippschaltung B1 ausgibt. Der Ausgang  $\overline{Q}$  der Kippschaltung B1 ist mit dem anderen Eingang der Verknüpfung ET5 verbunden. Der Ausgang der Verknüpfung ET5 ist mit dem Eingang load des LATCH-Registers verbunden.

Dieser Arbiter funktioniert wie folgt: ist

15 keine Abfrage auf den Leitungen  $\overline{rqst}$  vorhanden, dann speichert die Kippschaltung B1 permanent den Zustand der inaktiven Leitung  $\overline{e0}$  und gibt so durch die Verknüpfung ET5 das Laden des LATCH-Registers frei.

Die Ankunft eines Signals  $\overline{rqst}$  bewirkt die

20 Blockierung des Taktgebers und die Aktivierung des Signals  $\overline{grnt}$ , das dem Signal  $\overline{rqst}$  zugeordnet ist, und zwar bis zur Inaktivierung dieses letzteren: der Arbiter bleibt während der gesamten Dauer der sich im Gang befindlichen Transaktion in seinem Zustand.

25 Der in Abbildung 13 dargestellte Abfrage-Verwaltungsprozessor PGU, bildet eine Schnittstelle zwischen dem Verarbeitungs-Prozessor CPU und:

- einerseits den verschiedenen Prozessoren, mit denen er Informationen austauschen muß: paralleler

30 Verwaltungsprozessor PGP, Serien-Verwaltungsprozessor PGS und Verzeichnis-Verwaltungsprozessor des Cache-Speichers PGR,

- andererseits dem Verwaltungsverzeichnis des Cache-Speichers RG und dem Cache-Speicher MC.

35 Der Verarbeitungs-Prozessor CPU löst durch Aktivierung des Signals  $\overline{as}$  ("address strobe") die Aktivität des Abfrage-Verwaltungsprozessors PGU aus. Dieses Signal gibt den Adreßbus  $\overline{adr\_CPU}$ , die Lesesignale

$\overline{r}_{CPU}$  und Schreibsignale  $\overline{w}_{CPU}$  sowie die Funktionsleitungen  $fc_{CPU}$  des Verarbeitungs-Prozessors CPU frei. Der Verarbeitungs-Prozessor CPU geht dann bis zum Quittieren der Abfrage durch das Signal  $\overline{dtack}_{CPU}$  in den

5 Wartezustand.

Das Signal  $\overline{as}$  ist mit dem Eingang eines Differenzierers D10 verbunden. Der Ausgang dieses Differenzierers ist an einen der drei Eingänge einer Verknüpfung ET12 angeschlossen, und die beiden anderen

10 Eingänge empfangen das Signal  $\overline{ack}_{US}$  beziehungsweise  $\overline{ack}_{UP}$ . Dieses letztere Signal wird auch an den Eingang  $\overline{R}$  einer Kippschaltung B13 ausgegeben. Der Eingang  $\overline{S}$  der Kippschaltung B13 empfängt den Ausgang einer Verknüpfung NET10. Der Ausgang der Verknüpfung ET12 ist mit dem

15 Eingang  $\overline{S}$  einer Kippschaltung B11, mit dem Eingang  $\overline{S}$  der Kippschaltung B10 und mit dem Eingang  $\overline{clear10}$  eines Schieberegisters SR10 verbunden. Der Ausgang  $\overline{Q}$  der Kippschaltung B11 liefert das Signal  $\overline{rqst}_{UR}$ . Die Kippschaltung B11 empfängt auf ihrem Eingang  $\overline{R}$  die

20 invertierte Phase  $\overline{theta13}$  und die Kippschaltung B10 die invertierte Phase  $\overline{theta11}$ . Der Ausgang Q der Kippschaltung B10 ist mit dem Serieneingang  $\overline{serial\_in10}$  des Registers SR10 verbunden. Das Schieberegister SR10 empfängt auf seinem Eingang  $\overline{clk10}$  das Taktgebersignal -h- und auf

25 seinem Freigabeeingang  $\overline{en10}$  das Signal  $\overline{grnt}_{UR}$ .

Die Aktivierung des Signals  $\overline{as}$  löst den Differenzierer D10 aus. Der durch diesen Differenzierer erzeugte Impuls geht durch die Verknüpfung ET12, versetzt die Kippschaltungen B10 und B11 durch ihren Eingang  $\overline{S}$  in

30 den Logikzustand Eins und führt auch das Zurücksetzen auf Null des Schieberegisters SR10 durch seinen Eingang  $\overline{clear10}$  durch.

Die Kippschaltung B10 und das Schieberegister SR10 bilden die Logikunterbaugruppe "Phasenverteiler"

35 DP\_U. Die Aktivierung dieses Phasenverteilers wird dadurch ausgelöst, daß die Kippschaltung B10 auf Eins gesetzt und das Schieberegister SR10 auf 0 zurückgesetzt wird. Ist das Schieberegister durch das Vorhandensein eines Nullzustands

auf seinem Eingang  $\overline{en10}$  freigegeben worden, dann bewirkt der nächste Impuls des Taktgebers  $h$  auf den Eingang  $clk$  des Schieberegisters die Verschiebung um einen Schritt des besagten Registers.

5           Der Logikzustand Eins der Kippschaltung B10 wird auf dem Ausgang  $\theta_{ta11}$  des Schieberegisters SR10 durch seinem Serieneingang  $serialin10$  weitergegeben. Der Ausgang  $\theta_{ta11}$ , als invertierte Phase  $\theta_{ta11}$  bezeichnet, setzt die Kippschaltung B10 durch ihren Eingang  $\overline{R}$  auf Null  
10 zurück. So wird bei jeder Aktivierung des Phasenverteilers DP\_U in das Schieberegister SR10 ein einziges Bit eingegeben. Bei jedem Impuls des Taktgebers  $h$  verschiebt sich dieses Bit in dem Schieberegister SR10 und erzeugt die aufeinanderfolgenden getrennten Phasen  $\theta_{ta11}$ ,  
15  $\theta_{ta12}$  und  $\theta_{ta13}$ .

Das Setzen in den Logikzustand Eins der Kippschaltung B11 bewirkt die Aktivierung des Signals  $\overline{rqst\_UR}$ . Dieses Signal wird an den Verzeichnis-Verwaltungsprozessor PGR gesandt. Dieser letztere wird  
20 durch Aktivierung des Signals  $\overline{grnt\_UR}$  sobald wie möglich den Zugriff auf das Verzeichnisverzeichnis RG und auf den Cache-Speicher MC geben; dieses Signal gibt alle Übergabepuffer BV10, BV11 und BV12 frei, die sich auf den Bussen des Verwaltungsregisters beziehungsweise auf den  
25 Bussen des Cache-Speichers befinden. Dieses Signal  $\overline{grnt\_UR}$  gibt ebenfalls den Phasenverteiler frei, der dann sequentiell die Phasen  $\theta_{ta11}$ ,  $\theta_{ta12}$  und  $\theta_{ta13}$  erzeugt.

Die Phase  $\theta_{ta11}$  entspricht einer  
30 Zeitverzögerung, die das Lesen des Stichworts des durch den Verarbeitungs-Prozessor CPU im Verzeichnisverzeichnis RG angeforderten Blocks ermöglicht, der durch das Rahmenfeld der Adresse  $adr\_CPU$  adressiert ist und mit dem Bus  $adr\_RG$  über die Übergabepuffer BV10 verbunden ist. Das  
35 Signal  $\overline{r\_RG}$  ist immer am Eingang eines Puffers BV10 aktiv, das Signal  $\overline{w\_RG}$  am Eingang eines Puffers BV10 immer inaktiv. Am Ende der Zeitverzögerung, wird dem Prozessor PGU das Stichwort über den Bus  $data\_RG$  gesandt. Der Teil

tag dieses Stichworts und das Freigabebit  $v$  werden an einen der Vergleichseingänge eines Komparators COMP10 ausgegeben, wobei der andere Eingang mit dem Teil tag der Adresse  $adr\_CPU$  verbunden ist. Das dem Freigabebit  $v$  gegenüberliegende Bit ist immer auf Eins. Der Komparator COMP10 wird durch Vorhandensein eines Zustands Eins auf seinem Eingang  $en11$  permanent freigegeben.

Die Zugriffsdauer auf das Verwaltungsverzeichnis RG und die Taktgeberfrequenz  $h$  verhalten sich so zueinander, daß am Ende der Phase  $\theta_{et11}$ , der Ausgang  $eg10$  des Komparators COMP10 positioniert wird und die Information "der angeforderte Block ist im Cache-Speicher vorhanden oder im Cache-Speicher nicht vorhanden" liefert.

Ist der angeforderte Block im Cache-Speicher MC ( $eg10 = 1$ ) vorhanden, dann liefert das Signal  $eg10$ , das an einen der beiden Eingänge der Verknüpfung ET10 ausgegeben wird, ein durch die Phase  $\theta_{et12}$  kalibriertes Signal, die Phase ist auf dem anderen Eingang der Verknüpfung ET10 vorhanden.

Dieses auf dem Ausgang der Verknüpfung ET10 vorhandene kalibrierte Signal ist mit den Eingängen der Verknüpfungen NET10, NET11 und NET12 verbunden.

Die Verknüpfung NET10 empfängt auf ihren Eingängen, außer dem Ausgang der Verknüpfung ET10, das invertierte Zustandsbit  $m$  aus dem Stichwort und das invertierte Schreib-Anforderungssignal  $\overline{w\_CPU}$ .

Die Aktivierung der Verknüpfung NET10 entspricht dem Zustand "Schreib-Anforderung eines Wortes in einen Block, der in dem Cache-Speicher vorhanden ist und der noch nicht modifiziert worden ist ( $m = 0$ )". Der Ausgang der Verknüpfung NET10 ist mit dem Eingang  $\overline{S}$  einer Kippschaltung B13 verbunden. Die Aktivierung der Verknüpfung NET10 setzt die Kippschaltung B13 in den Logikzustand Eins, was durch die Leitung  $\overline{rqst\_UP}$  eine Informations-Schreib-Abfrage beim Verwaltungsprozessor des parallelen Drahts PGP auslöst. Die betreffende

Blockadresse wird über die Leitungen  $\text{adr\_bloc\_UP}$  geliefert, eine Abzweigung der Leitungen  $\text{adr\_CPU}$ .

Der Abfrage-Verwaltungsprozessor PGU hat den ersten Teil seiner Aufgabe beendet: das  
 5 Verwaltungsverzeichnis RG und der Cache-Speicher werden durch Inaktivierung des Signals  $\overline{\text{rqst\_UR}}$  freigemacht, die Folge des Empfangs der invertierten Phase  $\text{theta13}$  auf dem Eingang  $\overline{\text{R}}$  der Kippschaltung B11.

Der Informations-Schreibmechanismus wird im  
 10 Abschnitt "Verwaltungsprozessor des parallelen Drahts PGP" beschrieben und bewirkt das Setzen des angeforderten Blocks in den "modifiziert"-Zustand ( $m = 1$ ) oder "ungültig"-Zustand ( $v = 0$ ). Es kann festgestellt werden, daß die Freimachung des Verwaltungsverzeichnisses RG und  
 15 des Cache-Speichers MC durch den Abfrage-Verwaltungsprozessor PGU notwendig ist, damit der Verwaltungsprozessor des parallelen Drahts PGP darauf Zugriff nehmen kann. Das Ende der Operation "Informations-Schreibvorgang" wird dem Abfrage-Verwaltungsprozessor PGU  
 20 durch Aktivierung des Signals  $\overline{\text{ack\_UP}}$  signalisiert, wodurch die Kippschaltung B13 auf Null zurückgesetzt wird und durch die Verknüpfung ET12 die Kippschaltung B11 und der Phasenverteiler aktiviert wird: der anfangs durch das Signal  $\overline{\text{as}}$  gestartete Zyklus läuft erneut ab, jedoch wird  
 25 die Sequenz, eine Folge der Aktivierung der Verknüpfung NET10, in diesem Abfragezyklus nicht ein zweites Mal durchlaufen.

Die Verknüpfung NET11 empfängt auf ihren Eingängen, außer dem Ausgang der Verknüpfung ET10, das  
 30 Zustandsbit  $m$  aus dem Stichwort und das invertierte Schreib-Anforderungssignal  $\overline{w\_CPU}$ .

Die Aktivierung der Verknüpfung NET11 entspricht dem Zustand "Schreib-Anforderung in einen im Cache-Speicher vorhandenen Block, der bereits modifiziert  
 35 worden ist".

Der Ausgang der Verknüpfung NET11 ist durch einen der Puffer BV11 hindurch mit dem Schreibsignal  $\overline{w\_MC}$  des Cache-Speichers MC verbunden. Dieses Signal ermöglicht

das Schreiben der auf dem Bus data\_MC vorhandenen Daten im Cache-Speicher an der auf dem Bus adr\_MC vorhandenen Adresse, wobei der Bus data\_MC mit dem Bus data\_CPU über die bidirektionalen Puffer BV12 und der Bus adr\_MC mit dem  
 5 Bus adr\_CPU über die Puffer BV11 verbunden sind. Die Aktivierungsrichtung dieser Puffer wird durch das Signal  $\overline{w\_MC}$  geliefert.

Der Ausgang der Verknüpfung NET11 ist ebenfalls mit einem der Eingänge der Verknüpfung ET11  
 10 verbunden, die so dem Verarbeitungs-Prozessor CPU das Signal  $\overline{d\_ack\_CPU}$  zurücksendet. Der Schreibvorgang in den Cache-Speicher erfolgt parallel zu der Aktivierung des Signals  $\overline{d\_ack\_CPU}$ , was den gewöhnlichen Spezifikationen der Verarbeitungs-Prozessoren entspricht.

15 Die Operation wird durch das Freimachen des Verwaltungsverzeichnisses RG und des Cache-Speichers MC beendet, und zwar durch Inaktivieren des Signals  $\overline{rqst\_UR}$ , eine Folge des Empfangs der invertierten Phase theta13 auf der Kippschaltung B11.

20 Die Verknüpfung NET12 empfängt auf ihren Eingängen, außer dem Ausgang der Verknüpfung ET10, das invertierte Lese-Anforderungssignal  $\overline{r\_CPU}$ . Die Aktivierung der Verknüpfung NET12 entspricht dem Zustand "Anforderung zum Lesen eines Worts in einem im Cache-Speicher  
 25 vorhandenen Block".

Die nachfolgenden Operationen sind identisch mit den vorangegangenen, wobei der einzige Unterschied das der Tranferrichtung der Daten auf den Bussen data\_CPU und data\_MC zugeordnete aktivierte Signal ( $\overline{r\_MC}$  anstatt  $\overline{w\_MC}$ )  
 30 ist.

Ist der angeforderte Block nicht im Cache-Speicher vorhanden ( $eg10 = 0$ ), dann liefert das invertierte Signal  $eg10$ , das mit einem der beiden Eingänge der Verknüpfung NET13 verbunden ist, ein durch die Phase  
 35 theta12 kalibriertes Signal; diese Phase ist auf dem anderen Eingang der Verknüpfung NET13 vorhanden. Der Ausgang der Verknüpfung NET13 ist mit dem Eingang  $\overline{S}$  der Kippschaltung B12 verbunden. Dieses kalibrierte Signal

setzt die Kippschaltung B12 auf Eins, was das Senden einer Dienstabfrage  $\overline{rqst\_US}$  zum Verwaltungsprozessor des Seriendrahts PGS bewirkt. Dieser Prozessor empfängt auf den Leitungen  $adr\_bloc\_US$  ebenfalls die abzufragende Blockadresse und auf den Leitungen  $w\_US$ ,  $r\_US$  und  $fc\_US$  die Art der Abfrage.

Der Abfrage-Verwaltungsprozessor PGU hat den ersten Teil seiner Aufgabe beendet: das Verwaltungsverzeichnis RG und der Cache-Speicher MC werden durch Inaktivierung der Leitung  $\overline{rqst\_UR}$  freigemacht, eine Folge des Empfangs der invertierten Phase  $\theta_{eta13}$  auf der Kippschaltung B11.

Der Update-Mechanismus des Cache-Speichers ist im Abschnitt "Verwaltungsprozessor der Serienverbindung PGS" beschrieben.

Man kann feststellen, daß das Freimachen des Verwaltungsverzeichnisses RG und des Cache-Speichers MC notwendig ist, damit der Verwaltungsprozessor der Serienverbindung PGS darauf Zugriff nehmen kann.

Das Update des Cache-Speichers wird dem Abfrageprozessor PGU durch Aktivierung des Signals  $\overline{ack\_US}$  signalisiert. Dieses Signal wird an den Eingang  $\overline{R}$  der Kippschaltung B12 und an den Eingang der Verknüpfung ET12 ausgegeben. Es setzt so die Kippschaltung B12 auf Null zurück und aktiviert durch die Verknüpfung ET12 die Kippschaltung B11 und den Phasenverteiler: der anfangs durch das Signal  $\overline{as}$  gestartete Zyklus wird erneut durchlaufen, jedoch dieses Mal aufgrund des Vorhandenseins des Blocks im Cache-Speicher mit Erfolg.

Der Serienverwaltungsprozessor PGS, der als Beispiel in Abbildung 14 dargestellt ist, hat die Aufgabe, die Serienverbindung LS zu verwalten und zu diesem Zweck die Anforderungen für den Blocktransfer zwischen dem Hauptspeicher RAM und dem Cache-Speicher MC und die entsprechenden Updates im Verwaltungsverzeichnis RG durchzuführen. Er verarbeitet an erster Stelle die Anforderungen aus dem Spionprozessor PE, die in einer

Warteschlange FIFO warten. Er verarbeitet ebenso die Anforderungen aus dem Abfrageprozessor PGU.

Dieser Verwaltungsprozessor der Serienverbindung PGS umfaßt eine Kippschaltung B20, die auf ihrem Dateneingang D das Signal empty aus der Warteschlange FIFO und auf ihrem Taktgebereingang den Ausgang  $\bar{Q}$  einer Kippschaltung B22 empfängt. Eine Kippschaltung B21 empfängt auf ihrem Dateneingang den Ausgang einer Verknüpfung OU20. Diese Verknüpfung OU20 gibt das Signal rqst\_US frei, das sie auf einem ihrer beiden Eingänge empfangen hat, wobei der andere Eingang mit dem Signal empty verbunden ist. Der Taktgebereingang der Kippschaltung B21 kommt aus dem Ausgang Q der Kippschaltung B22. Der Ausgang  $\bar{Q}$  der Kippschaltung B22 ist an ihrem Dateneingang D zur Schleife geschaltet, was sie als Frequenzteiler durch zwei bedingt. Der Taktgebereingang der Kippschaltung B22 ist an den Ausgang einer Verknüpfung ET20 angeschlossen, die auf einem ihrer Eingänge den allgemeinen Betriebstaktgeber h und auf dem anderen Eingang ein Freigabesignal empfängt. Dieses Freigabesignal kommt vom Ausgang einer Verknüpfung ET24, die auf ihren beiden Eingängen den Ausgang  $\bar{Q}$  beziehungsweise Q der Kippschaltungen B20 und B21 empfängt.

Die Kippschaltung B20 empfängt auf ihrem Eingang  $\bar{R}$  die invertierte Phase theta25 aus einem Phasenverteiler DP\_S. Die Kippschaltung B21 empfängt auf ihrem Eingang  $\bar{S}$  den Ausgang einer Verknüpfung NOU20. Diese Verknüpfung NOU20 empfängt auf ihren beiden Eingängen den Ausgang der Verknüpfung ET22 beziehungsweise ET23. Die Verknüpfung ET22 empfängt auf ihren Eingängen die Phase theta25 aus dem Phasenverteiler und das Signal maj aus einer Verknüpfung ET29. Die Verknüpfung ET23 empfängt auf ihren Eingängen die Phase theta27 aus dem Phasenverteiler und das invertierte Signal maj.

Die Bauteile B20, B21, B22, ET20, ET22, ET23, OU20, ET24 und NOU20 bilden zusammen einen Arbiter mit einer festen Priorität ARB\_S. Er funktioniert wie folgt:

die Kippschaltung B22 liefert auf ihren Ausgängen  $\overline{Q}$  und  $Q$  wechselnde Signale mit einer Frequenz, die halb so groß ist, wie die des allgemeinen Taktgebers. Diese Signale geben abwechselnd die Kippschaltungen B20 und B21 frei.

5 Ist eine Dienstabfrage auf einem der Eingänge der Kippschaltungen B20 oder B21 vorhanden, dann speichern diese wechselnden Signale die Anforderung in der entsprechenden Kippschaltung (B20 für den Spionprozessor PE und B21 für den Abfrage-Verwaltungsprozessor PGU), die

10 daraufhin den wechselnden Betrieb blockiert. Es ist festzustellen, daß das Signal  $\overline{rqst\_US}$ , das vom Abfrage-Verwaltungsprozessor PGU kommt, durch das Signal  $\overline{empty}$  über die Verknüpfung OU20 bedingt wird: dieses Signal wird dann nur in Betracht bezogen, wenn die Warteschlange FIFO

15 leer ist. Die Kippschaltung B20 (beziehungsweise B21) wird auf Null nur zurückgesetzt, wenn die auszuführende Transaktion beendet ist.

Die Verteilung einer Anforderung auf der einen oder anderen der beiden Kippschaltungen B20 und B21 zeigt

20 sich durch eine Zustandsänderung des Ausgangs der Verknüpfung ET24. Der Ausgang der Verknüpfung ET24 ist ebenfalls mit einem Differenzierer D20 verbunden, der während der Zustandsänderung des Ausgangs der Verknüpfung ET24 einen Impuls ausgibt. Der Ausgang des Differenzierers

25 ist einerseits mit dem Phasenverteiler DP\_S (Eingang  $\overline{S}$  einer Kippschaltung B36 und  $\overline{clr20}$  eines Schieberegisters SR20) des Verwaltungsprozessors der Serienverbindung verbunden und andererseits mit einem der beiden Eingänge der Verknüpfung OU22. Der Ausgang der Verknüpfung ist mit

30 den Eingängen  $\overline{S}$  der beiden Kippschaltungen B24 und B25 verbunden. Die Kippschaltung B24 empfängt auf ihrem Eingang  $\overline{R}$  den Ausgang einer Verknüpfung NOU21 und die Kippschaltung B25 empfängt das Signal  $\overline{grnt\_SR}$ . Die Verknüpfung NOU21 empfängt auf ihren beiden Eingängen die

35 Ausgänge von Verknüpfungen ET36 und ET37. Die Verknüpfung E36 empfängt auf ihren Eingängen die Phase  $\theta_{23}$ , die aus dem Phasenverteiler kommt, das Signal  $maj$ , die

Verknüpfungen B37, die Phase theta27, die aus dem Phasenverteiler kommt, und das invertierte Signal maj.

Der aus dem Differenzierer D20 kommende Impuls initialisiert so den Phasenverteiler und setzt eine der Kippschaltungen B24 und B25 über die Verknüpfung OU22 in den Logikzustand Eins.

Der Phasenverteiler DP\_S setzt sich aus dem Schieberegister SR20 und der Kippschaltung B36 zusammen. Er funktioniert genauso wie der im Abschnitt über den Abfrage-Verwaltungsprozessor PGU beschriebene.

Der Ausgang  $\bar{Q}$  der Kippschaltung B24 ist mit dem Signal  $\overline{rqst\_SR}$ , das für den Verzeichnis-Verwaltungsprozessor PGR bestimmt ist, verbunden. Seine Aktivierung löst eine Dienstanforderung an diesen Prozessor aus, der über die Leitung  $\overline{grnt\_SR}$  antwortet, die ihrerseits mit dem Eingang  $\bar{R}$  der Kippschaltung B25 verbunden ist. Der Ausgang Q der Kippschaltung B25 ist mit einem der Eingänge einer Verknüpfung OU23 verbunden. Der Ausgang der Verknüpfung OU23 ist mit dem Eingang  $\overline{en20}$  des Schieberegisters SR20 verbunden.

Die Logikgruppe B24 und B25 bilden zusammen eine Logik RESYNC\_S für die erneute Synchronisation zwischen asynchronen Einheiten. Sie funktioniert folgendermaßen:

Eine Dienstanforderung  $\overline{rqst\_SR}$  an den Verzeichnis-Verwaltungsprozessor PGR erfolgt durch Aktivierung der Kippschaltungen B24 und B25 über die Verknüpfung OU22, was zwei Aktivierungsquellen zuläßt. Die Logik des Verzeichnis-Verwaltungsprozessors PGR sorgt innerhalb einer nicht bestimmten Zeitdauer für eine Antwort  $\overline{grnt\_SR}$ , die die Kippschaltung B25 auf Null zurücksetzt. Die Kippschaltung B24 hält bis zu ihrem Zurücksetzen auf Null ihre Anforderung durch Aktivierung ihres Eingangs  $\bar{R}$  aufrecht. Dafür inaktiviert der Verzeichnis-Verwaltungsprozessor PGR seine Leitung  $\overline{grnt\_SR}$ : die Resynchronisationslogik ist bereit für eine nächste Dienstanforderung. Der Ausgang Q der Kippschaltung B25 dient dazu, den Phasenverteiler durch Betätigen seines

Eingangs  $\overline{en20}$  über die Verknüpfung OU23 zu blockieren: das Zurücksetzen auf Null der Kippschaltung B25 macht den Phasenverteiler frei, der die erste Phase  $\theta_{21}$  während des nächsten aktiven Übergangs des allgemeinen Taktgebers h, der an den Eingang  $clk20$  des Phasenverters  
 5 h, der an den Eingang  $clk20$  des Phasenverters angeschlossen ist, liefert.

Die Leitung  $\overline{grnt\_SR}$  ist ebenfalls mit den Freigabepuffern BV20, BV25 und BV21, BV22 verbunden, die den Zugriff zum Verwaltungsverzeichnis RG beziehungsweise  
 10 zum Cache-Speicher MC öffnen.

Ist die Kippschaltung B20 aktiv, dann ist die sich im Gang befindliche Transaktion ein vollständiges Entladen des Blocks, die vom Spionprozessor PE über die Warteschlange FIFO angefordert worden ist. Der Ausgang  $\overline{Q}$   
 15 der Kippschaltung B20 ist mit den Freigabepuffern BV24 verbunden. Diese Puffer empfangen auf ihrem Eingang den Ausgang eines Registers REG20. Der Ausgang  $\overline{Q}$  der Kippschaltung B20 ist mit einem der beiden Eingänge einer Verknüpfung OU21 verbunden, die auf ihrem anderen Eingang  
 20 den Ausgang des Differenzierers D20 empfängt. Der Ausgang der Verknüpfung OU21 ist mit dem Eingang  $\overline{load20}$  des Registers REG20 und dem Eingang  $\overline{read20}$  der Warteschlange FIFO verbunden.

So bewirkt die Aktivierung der Kippschaltung  
 25 B20:

1. Die Initialisierung des Phasenverters,
2. Eine Anforderung für den Zugriff auf das Verwaltungsverzeichnis RG und auf den Cache-Speicher MC,
3. Das Laden des ersten Elements der  
 30 Warteschlange FIFO in das Register REG20 und das Vorrücken der Warteschlange,
4. Die Freigabe des Puffers BV24: der Bus  $adr\_X$  enthält die zu entladende Blockadresse. Die Art der Operation (Update) wird anhand der Kombination der Bits v  
 35 und m (Signal maj) gefunden.

Ist die Kippschaltung B21 aktiv, dann stammt die sich im Gang befindliche Transaktion aus einem Informationsmangel im Cache-Speicher MC. Der Ausgang  $\overline{Q}$  der

Kippschaltung B21 ist mit Freigabepuffern BV23 verbunden. Diese Puffer empfangen auf ihrem Eingang die vom Abfrage-Verwaltungsprozessor PGU stammenden Informationen.

So bewirkt die Aktivierung der Kippschaltung  
5 B21 folgendes:

1. Die Initialisierung des Phasenverteilers,
2. Eine Anforderung für den Zugriff auf das  
Verwaltungsverzeichnis RG und zum Cache-Speicher MC,
3. Die Freigabe der Puffer BV23: der Bus adr\_X  
10 enthält die Blockadresse, die den Informationsmangel im  
Cache-Speicher MC bewirkt hat, und die Art der Abfrage:  
Lesen oder Schreiben, geteilte oder nicht geteilte Daten  
(Leitungen fc\_US).

Das Feld "Rahmen" des Busses adr\_X ist über  
15 die Freigabepuffer BV20 mit den Adressenleitungen adr\_RG  
des Verwaltungsverzeichnisses RG verbunden. Die Ausgänge  $\bar{Q}$   
der Kippschaltungen B26 und B27 sind über die  
Freigabepuffer BV20 mit der Leitung  $\bar{r}_{RG}$  beziehungsweise  
 $\bar{w}_{RG}$  des Verwaltungsverzeichnisses verbunden. Die  
20 Kippschaltung B26 empfängt auf ihrem Eingang  $\bar{S}$  den Ausgang  
der Verknüpfung OU22 und auf ihrem Eingang  $\bar{R}$  die  
invertierte Phase theta22, die aus dem Phasenverteiler  
kommt. Die Kippschaltung B27 empfängt auf ihrem Eingang  $\bar{S}$   
den Ausgang einer Verknüpfung NOU22 und auf ihrem Eingang  
25  $\bar{R}$  den Ausgang einer Verknüpfung NOU23. Die Verknüpfung  
NOU22 empfängt auf ihren beiden Eingängen den Ausgang der  
Verknüpfung ET25 beziehungsweise ET26, die ihrerseits auf  
ihren Eingängen, was die Verknüpfung ET25 betrifft, die  
Phase theta22 und das Signal maj empfangen und, was die  
30 Verknüpfung ET26 betrifft, die Phase theta26 und das  
Signal maj. Die Verknüpfung NOU23 empfängt auf ihren  
beiden Eingängen den Ausgang der Verknüpfung ET27  
beziehungsweise ET28, die ihrerseits auf ihren Eingängen,  
was die Verknüpfung ET27 betrifft, die Phase theta23 und  
35 das invertierte Signal maj empfangen und, was die  
Verknüpfung ET28 betrifft, die Phase theta27 und das  
invertierte Signal maj.

Das Feld tag des Busses adr\_X ist über Freigabepuffer BV25 mit dem Feld tag der Leitungen data\_RG verbunden. Diese letzteren empfangen auf ihrem Freigabeeingang den Ausgang einer Verknüpfung OU24, die  
 5 auf ihren Eingängen das Signal  $\overline{\text{grnt\_SR}}$  und den Ausgang  $\overline{Q}$  der Kippschaltung B27 empfängt.

Die Eingänge D von Kippschaltungen B28 und B29 sind mit der Leitung des Freigabebits v beziehungsweise des Modifikationsbits m des Busses data\_RG verbunden. Die  
 10 Taktgebereingänge der Kippschaltungen B28 und B29 sind mit der Phase theta22 des Phasenverteilers verbunden. Der Ausgang Q der Kippschaltung B28 und der Ausgang Q der Kippschaltung B29 sind mit den beiden Eingängen einer Verknüpfung ET29 verbunden, die auf ihrem Ausgang das  
 15 Signal maj liefert. Eine Verknüpfung OU25 empfängt auf ihren Eingängen das Signal maj und den Ausgang  $\overline{Q}$  einer Kippschaltung B30. Der Ausgang der Verknüpfung OU25 ist mit dem Auswahl Eingang sel20 eines Multiplexers MUX20 verbunden, der auf seinen beiden Dateneingängen den  
 20 Ausgang Q der Kippschaltung B28 (Bit v) und die Konstante 0 empfängt, wobei die Konstante Null gewählt wird, wenn sich der Auswahl Eingang sel20 im Logikzustand Eins befindet. Der Ausgang des Multiplexers MUX20 ist mit der Leitung des Freigabebits v des Busses adr\_X verbunden. Die  
 25 Leitung des Wartebits -a- des Busses adr\_X wird in den Logikzustand Null gesetzt. Das Modifikationsbit -m- ist mit der Lese-Leitung  $\overline{r\_adr\_x}$  des Busses adr\_X verbunden.

Die gesamte oben beschriebene Logik bildet die Logik für den Zugriff und das Update des  
 30 Verwaltungsverzeichnisses RG. Sie funktioniert wie folgt: die Aktivierung des Signals  $\overline{\text{grnt\_SR}}$ , die den Zugriff auf das Verwaltungsverzeichnis und den Cache-Speicher erlaubt, gibt die Freigabepuffer BV20 frei. Das Lesen des betreffenden Stichworts wird vom Anfang der  
 35 Zugriffserlaubnis bis zur Ankunft der Phase theta22 gesteuert, die dem Moment der Einspeicherung der Bits -v- und -m- in die Kippschaltungen B28 und B29 entspricht. Der kombinierte Zustand dieser beiden Bits erzeugt durch die

Verknüpfungen ET29 das Signal maj, das den Fortlauf der Operationen bedingt.

1. Fall: maj = 1. Dieser Fall kommt vor, wenn sowohl das Freigabebit als auch das Modifikationsbit den Wert 1 hat. Dieser Fall entspricht entweder einer Anforderung nach vollständigem Entladen des Blocks des Spionprozessors PE oder aber einem Informationsmangel, der vom Abfrage-Verwaltungsprozessor PGU an einer belegten und modifizierten Blockstelle festgestellt wird: in beiden Fällen muß der betreffende Block in den Hauptspeicher RAM eingeschrieben werden.

Dazu ist das Feld -Rahmen- des Busses adr\_X über die Freigabepuffer BV21 mit den Leitungen adr\_MC verbunden. Die Ausgänge  $\bar{Q}$  der Kippschaltungen B30 und B31 sind mit der Leitung  $\bar{F}_{MC}$  beziehungsweise  $\bar{W}_{MC}$  des Cache-Speichers MC verbunden, und zwar durch die Freigabepuffer BV21. Die Leitung  $\bar{bloc}$  wird über einen der Freigabepuffer BV21 auf Null gesetzt. Die Datenleitungen data\_MC des Cache-Speichers sind über die bidirektionalen Freigabepuffer BV22 mit den Eingängen des Schieberegisters RDP und mit den Ausgängen der Freigabepuffer BV26 verbunden, die auf ihren Eingängen die Ausgangsleitungen des Schieberegisters RDP empfangen. Die Puffer BV22 werden durch die Leitung  $\bar{grnt\_SR}$  freigegeben und ihre Freigaberichtung wird durch den Ausgang  $\bar{Q}$  der Kippschaltung B31 gesteuert. Die Kippschaltung B30 empfängt auf ihren Eingängen  $\bar{S}$  und  $\bar{R}$  die invertierte Phase theta21 beziehungsweise theta23, die aus dem Phasenverteiler kommen. Die Kippschaltung B31 empfängt auf ihren Eingängen  $\bar{S}$  und  $\bar{R}$  die Ausgänge der Verknüpfung ET32 beziehungsweise ET33. Die Verknüpfung ET32 empfängt auf ihren Eingängen die Phase theta26 und das invertierte Signal maj, die Verknüpfungen ET33 die Phase theta27 und das invertierte Signal maj. Eine Verknüpfung NET20 empfängt auf ihren Eingängen die Phase theta23 und das Signal maj. Der Ausgang einer Verknüpfung ET35 steuert die Freigabepuffer BV26 und empfängt auf ihren beiden Eingängen das invertierte Signal maj beziehungsweise den

Ausgang Q der Kippschaltung B31. Der Ausgang der Verknüpfung NET20 ist mit dem Signal load21 des Schieberegisters RDP und mit dem Eingang  $\bar{S}$  einer Kippschaltung B32 verbunden. Der Eingang  $\bar{R}$  dieser Kippschaltung B32 empfängt das Signal fin\_transfert\_maj aus der Logik TFR, die dem Schieberegister RDP zugeordnet ist. Der Ausgang Q der Kippschaltung B32 ist mit einem der Eingänge der Verknüpfung OU23 verbunden.

Die oben beschriebene Logik ermöglicht das vollständige Entladen eines Blocks des Cache-Speichers. Es funktioniert folgendermaßen:

Parallel zu dem Zugriff auf das Verwaltungsverzeichnis RG, wird durch die Leitung  $\bar{r}_{MC}$ , die aus der Kippschaltung B30 kommt, ein Lesevorgang des Cache-Speichers MC aktiviert, und zwar während der Phasen theta21 und theta22. Am Ende dieses Lesevorgangs werden die gelesenen Daten, die den zu entladenden Block darstellen, am Eingang des Schieberegisters RDM vorgelegt. Die Aktivierung des Signals maj, dessen Zustand zu Anfang der Phase theta22 bekannt ist, bewirkt folgendes:

1. Das Ungültigmachen des Blocks im Verwaltungsverzeichnis: da sich der Eingang sel20 des Multiplexers MUX20 im Logikzustand Eins befindet, wird das Freigabebit auf Null gesetzt und das Stichwort wird bei der Aktivierung des Signals  $\bar{w}_{RG}$ , die durch die Kippschaltung B27 während des Zyklus theta22 gesteuert wird, in das Verwaltungsverzeichnis RG geschrieben.

2. Das Laden des Schieberegisters RDM und die Transferaktivierung während des Übergangs von der Phase theta22 zur Phase theta23,

3. Das Setzen der Kippschaltung B32 in den Logikzustand Eins, das den Phasenverteiler bei der Phase theta23 bis zum Ende des Transfers blockiert, das durch das Signal fin\_transfert\_maj aus der Logik TFR signalisiert wird,

4. Das Freimachen des Verwaltungsverzeichnisses RG und des Cache-Speichers MC

durch Zurücksetzen der Kippschaltung B24 auf Null bei der Phase theta23.

So wird der Zugriff auf das Verzeichnis freige-  
 gemacht (somit zugänglich für den Spionprozessor PE), der  
 5 Update-Transfer ist im Gange und der Phasenverteiler auf  
 theta23 blockiert.

Sobald der Transfer beendet ist, wartet der  
 Block im Schieberegister RDM und es muß der  
 Verwaltungsprozessor des parallelen Drahts aktiviert  
 10 werden, damit das Schreiben in den Hauptspeicher RAM  
 effektiv ist.

Dazu ist die Kippschaltung B33 durch ihren  
 Ausgang Q mit der Dienstabfrageleitung  $\overline{rqst\_SP}$  verbunden,  
 die für den Verwaltungsprozessor des parallelen Drahts PGP  
 15 bestimmt ist. Der Ausgang Q ist mit einem der Eingänge der  
 Verknüpfung OU23 verbunden, der Eingang  $\overline{S}$  mit der  
 invertierten Phase theta24 des Phasenverters und der  
 Eingang  $\overline{R}$  mit dem Signal  $\overline{ack\_SP}$ . Der Bus  $adr\_X$  ist mit dem  
 Bus  $adr\_bloc\_SP$  des Verwaltungsprozessors des parallelen  
 20 Drahts PGP verbunden. Eine der Busleitungen  $adr\_bloc\_SP$   
 empfängt das Signal  $maj$ , um die Abfrageart anzugeben:  
 Update.

Sobald der Phasenverteiler durch das Signal  
 $\overline{fin\_transfert\_maj}$  freigemacht wird, bewirkt der  
 25 darauffolgende aktive Übergang des allgemeinen Taktgebers  
 h den Wechsel der Phase theta23 zur Phase theta24. Die  
 Phase theta24 bewirkt eine Dienstabfrage beim  
 Verwaltungsprozessor des parallelen Drahts PGP  
 (Aktivierung der Leitung  $\overline{rqst\_SP}$ ) und das Blockieren des  
 30 Phasenverters bis zum Quittieren der Abfrage durch das  
 Signal  $\overline{ack\_SP}$ . In dem Moment wird der Update-  
 Schreibvorgang durch den Verwaltungsprozessor des  
 parallelen Drahts PGP tatsächlich durchgeführt worden  
 sein. Der Phasenverteiler wird dann bei dem nächsten  
 35 aktiven Übergang des Taktgebers h von der Phase theta24  
 zur Phase theta25 übergehen.

Das Update des Hauptspeichers RAM ist beendet:  
 die Kippschaltung B20 wird durch Aktivierung ihres

Eingangs  $\bar{R}$  durch die invertierte Phase theta25 auf Null gesetzt. Die Kippschaltung B21 wird durch Aktivierung ihres Eingangs  $\bar{S}$  durch die Phase theta25, die durch das Signal maj mittels der Verknüpfung ET22 über die Verknüpfungen NOU22 bedingt ist, auf Eins gesetzt. Im Falle eines Informationsmangels im Cache-Speicher, stellt das vollständige Entladen eines Blocks nur den ersten Teil der Abfrage dar: die Anforderung  $\overline{rqst\_US}$  ist immer noch vorhanden, jedoch ermöglicht das Freimachen der Kippschaltung B21 die Inbetrachtziehung eventueller Update-Anforderungen, die sich in der Warteschlange FIFO im Wartezustand befinden. Sobald die Warteschlange FIFO leer ist (empty = 0), wiederholt sich der gesamte zuvor beschriebene Zyklus, dieses Mal mit dem Freigabebit auf Null. Man befindet sich dann in dem nachfolgenden Fall.

2. Fall: maj = 0. Dieser Fall kommt vor, wenn das Freigabebit auf Null ist (eventuell infolge eines vollständigen Entladens eines Blocks) oder wenn das Freigabebit auf Eins ist, das Modifikationsbit jedoch auf Null: die aktualisierte Kopie dieses Blocks befindet sich bereits im Hauptspeicher RAM.

So bewirkt die Dienstabfrage  $\overline{rqst\_SR}$  dafür eine Zugriffserlaubnis zum Verwaltungsverzeichnis RG und zum Cache-Speicher MC mit Lesen des Stichworts, Einspeichern der Bits m und v, Erzeugung des Signals maj, erneutes Schreiben des Stichworts mit v = 0 (maj befindet sich im Null-Zustand, die Kippschaltung B31 befindet sich im Null-Zustand und ihr Ausgang Q hat also den Wert Eins, was auf dem Eingang sel20 des Multiplexers MUX20 ein Logiksignal im Zustand Eins fördert und somit die Konstante 0 setzt) und das Freimachen des Zugangs zum Verwaltungsverzeichnis RG und zum Cache-Speicher MC. Diese Operationen werden wirksam, sobald die Phase theta23 aktiviert worden ist.

Der angeforderte Block muß nun im Hauptspeicher RAM gelesen werden. Dazu wird das invertierte Signal maj an einem der Eingänge einer Verknüpfung NET21 empfangen, die auf ihrem anderen Eingang

die Phase theta25 empfängt. Der Ausgang der Verknüpfung NET21 ist mit den Eingängen  $\bar{S}$  der Kippschaltungen B34 und B35 verbunden. Die Kippschaltung B34 empfängt auf ihrem Eingang  $\bar{R}$  das Signal fin\_réception, das von der  
 5 Transferlogik TFR kommt. Ihr Ausgang Q ist mit dem Differenzierer D21 verbunden. Dieser Differenzierer ist mit der Verknüpfung OU22 verbunden. Der Eingang  $\bar{R}$  der Kippschaltung B35 ist mit dem Signal grnt\_SR und der Ausgang Q mit einem der Eingänge der Verknüpfung OU23  
 10 verbunden.

Das Lesen eines Blocks im Hauptspeicher RAM und sein Transfer in den Cache-Speicher werden folgendermaßen durchgeführt: der Übergang der Phase theta23 zur Phase theta24 löst eine Dienstabfrage beim  
 15 Verwaltungsprozessor des parallelen Drahts aus, und zwar durch Aktivierung der Leitung rqst\_SP, die von der Kippschaltung B33 kommt. Die Art der Operation ist dieses Mal ein Lesevorgang (r\_adr\_X = 0) oder ein Schreibvorgang (w\_adr\_X = 0) und der Bus adr\_X liefert auf dem Bus  
 20 adr\_bloc\_SP die Adresse des angeforderten Blocks. Der Phasenverteiler ist bis zur Ankunft des Quittiersignals ack\_SP blockiert: die Lese- oder Schreib-Anforderung ist durch den Verwaltungsprozessor des parallelen Drahts PGP an den Hauptspeicher RAM gemacht worden, und der Block ist  
 25 durch diesen selben Prozessor zur gleichen Zeit freigegeben und mit "im Wartezustand" gekennzeichnet worden. Der Transfer vom Hauptspeicher RAM zum Schieberegister RDP ist also im Gange.

Der freigemachte Phasenverteiler liefert dann  
 30 die Phase theta25. Diese Phase setzt mittels der Verknüpfung NET21, die auf ihrem anderen Eingang das invertierte Signal maj empfängt, die Kippschaltungen B34 und B35 in den Logikzustand Eins. Die Kippschaltung B35 blockiert den Phasenverteiler. Die Kippschaltung B34 wird  
 35 bei Ankunft des im Register RDP angeforderten Blocks, die durch die Aktivierung der Leitung fin\_réception aus der Transferlogik TFR signalisiert wird, auf Null zurückgesetzt, was durch Aktivierung der Leitung rqst\_SR

aus der Kippschaltung B24 die Aktivierung des Differenzierers D21 und eine Zugriffsanforderung auf das Verwaltungsverzeichnis RG und auf den Cache-Speicher MC bewirkt.

- 5 Die Zugriffserlaubnis, die durch die Leitung  $\overline{\text{grnt\_SR}}$  signalisiert wird, entblockt den Phasenverteiler durch Zurücksetzen der Kippschaltungen B25 und B35 auf Null und öffnet den Zugang zum Verwaltungsverzeichnis RG und zum Cache-Speicher MC. Der Phasenverteiler liefert  
10 beim darauffolgenden aktiven Übergang des allgemeinen Taktgebers h die Phase  $\theta_{26}$ .

Die Kippschaltung B27, die mit dem Signal  $\overline{w\_RG}$  des Verwaltungsverzeichnisses verbunden ist, ist von  $\theta_{26}$  bis  $\theta_{27}$  aktiv, was die Durchführung des  
15 Updates des Verwaltungsverzeichnisses ermöglicht, mit:

- Feld tag des Busses  $\text{data\_RG}$  = Feld tag des Busses  $\text{adr\_X}$ ,

- Freigabebit  $v = 1$  ( $\text{maj} = 0$  und der Ausgang Q der Kippschaltung B31 ist auf Null: der Multiplexer MUX20  
20 läßt  $v$  durch, das durch den Verwaltungsprozessor des parallelen Drahts PGP auf Eins gesetzt worden ist oder durch den Spionprozessor PE bereits auf Null zurückgesetzt worden ist),

- Modifikationsbit  $m$  = Zustand der Leitung  
25  $\overline{r\_adr\_X}$  des Busses  $\text{adr\_X}$  (Schreib-Anforderung bewirkt  $m = 1$ , Lese-Anforderung  $m = 0$ ),

- Transfer-Wartebit  $a = 0$ ,

- Bit  $f = 0$ .

Die Kippschaltung B31 aktiviert das Signal  
30  $\overline{w\_MC}$  des Cache-Speichers MC von  $\theta_{26}$  bis  $\theta_{27}$ , was das Schreiben des Inhalts des Schieberegisters RDM (die Puffer BV26 werden durch den Ausgang  $\overline{Q}$  der Kippschaltung B31 und das invertierte Signal  $\text{maj}$  freigegeben) über die Freigabepuffer BV22, die von der Kippschaltung B31 für die  
35 Transferrichtung gesteuert werden, an der richtigen Stelle (Feld -Rahmen- des Busses  $\text{adr\_X}$ , verbunden mit dem Bus  $\text{adr\_MC}$ ) des Cache-Speichers ermöglicht.

Bei Ankunft der Phase theta27, ist das Update des Verwaltungsverzeichnisses und des Cache-Speichers beendet. Die Ankunft der Phase theta27 bewirkt das Zurücksetzen der Kippschaltung B24 auf Null, was den  
 5 Zugang zum Verwaltungsverzeichnis RG und zum Cache-Speicher MC freimacht, und die Aktivierung des Ausgangs der Verknüpfung ET23, was die Kippschaltung B21 auf Eins zurücksetzt und das Signal  $\overline{\text{ack\_US}}$  für den Abfrage-Verwaltungsprozessor PGU aktiviert: die Abfrage ist  
 10 beendet.

Im übrigen hat der Spionprozessor PE die Aufgabe der Aufrechterhaltung der Kohärenz der Daten im Cache-Speicher MC; als Beispiel ist in Abbildung 15 eine Architektur dieses Prozessors dargestellt. Dieser wird bei  
 15 jedem aktiven Übergang des Signals  $\overline{\text{valid}}$  des gemeinsamen Busses BUSA ausgelöst.

Wird das Signal  $\overline{\text{valid}}$  durch einen anderen als den dem Spionprozessor PE zugeordneten Verwaltungsprozessor des parallelen Drahts PGP aktiviert,  
 20 dann hat der Spionprozessor folgende Aufgabe, die abhängig von der Abfrageart ist:

- Lese-Abfrage eines Blocks von nicht geteilten Daten oder Schreib-Abfrage eines Block-Updates:  
 keine,
- 25 - Schreib-Abfrage eines Blocks von geteilten Daten:
  - . Block nicht vorhanden: keine,
  - . Block vorhanden und nicht modifiziert:  
 keine,
  - 30 . Block vorhanden und modifiziert:  
 Anforderung nach einem vollständigen Entladen des Blocks (ob dieser Block vorhanden ist oder sich bei einem Transfer vom Hauptspeicher RAM zum Cache-Speicher MC befindet, der Zustand wird vom Bit -a- des Stichworts  
 35 signalisiert),
  - Schreib-Abfrage eines Blocks von geteilten Daten:
    - . Block nicht vorhanden: keine,

- . Block vorhanden und nicht modifiziert:  
Block ungültig machen,
- . Block vorhanden und modifiziert:  
Anforderung nach vollständigem Entladen des Blocks  
5 (dieselbe Anmerkung wie oben),
  - Informations-Schreib-Abfrage eines Blocks:
    - . Block nicht vorhanden: keine,
    - . Block vorhanden und nicht modifiziert:  
Block ungültig machen,
    - 10 . Block vorhanden und modifiziert: Fall ist unmöglich,
  - . befindet sich eine Informations-Schreib-Anforderung auf demselben Block im Wartezustand, so wird sie in eine Block-Schreib-Abfrage umgewandelt, da dieser  
15 Block ungültig gemacht worden ist; dazu wird die sich im Gang befindliche Abfrage annulliert und ein Quittieren an den Abfrage-Verwaltungsprozessor PGU zurückgesandt. Dieser letztere konsultiert das Verzeichnis RG und findet dort den nicht vorhandenen Block: eine Abfrage wird an den  
20 Verwaltungsprozessor PGS gesandt. Diese Operation wird von dem parallelen Verwaltungsprozessor PGP in Betracht gezogen.
- Wird von dem dem Spionprozessor zugeordneten Verwaltungsprozessor des parallelen Drahts PGP das Signal  
25 valid aktiviert, dann hat der Spionprozessor folgende Aufgabe, die abhängig von der Abfrageart (der sogenannten lokalen Abfrage) ist:
  - Lese-Abfrage eines Blocks von nicht geteilten Daten oder Schreib-Abfrage eines Block-Updates:  
30 keine,
  - Lese-Abfrage eines Blocks von geteilten Daten: den Block als gültig, im Transferwartezustand und als nicht modifiziert ( $v = a = 1$  und  $m = 0$ ) kennzeichnen,
  - Schreib-Abfrage eines Blocks von geteilten  
35 Daten: den Block als gültig, im Transferwartezustand und als modifiziert ( $v = m = a = 1$ ) kennzeichnen,
  - Informations-Schreib-Abfrage eines Blocks: den Block mit "modifiziert" ( $m = 1$ ) kennzeichnen.

Um für die Durchführung der oben beschriebenen Funktionen zu sorgen, umfaßt der Spionprozessor PE einen Phasenverteiler DP\_E, der aus einem Schieberegister SR40 und einer Kippschaltung B40 besteht. Der Ausgang eines

5 Differenzierers D40 ist mit dem Eingang  $\overline{S}$  der Kippschaltung B40 und dem Eingang  $\overline{clear40}$  des Registers SR40 sowie mit dem Eingang  $\overline{S}$  einer Kippschaltung B41 verbunden. Der Ausgang Q der Kippschaltung B40 ist mit dem Eingang  $\overline{serial\_in40}$  des Registers SR40 verbunden. Der

10 Eingang  $\overline{clk40}$  des Registers SR40 empfängt das Signal h des allgemeinen Taktgebers und der Freigabeeingang  $\overline{en40}$  ist mit dem Ausgang Q der Kippschaltung B43 verbunden. Die invertierte Phase  $\overline{theta41}$  aus dem Register SR40 ist mit dem Eingang  $\overline{R}$  der Kippschaltung B40 verbunden.

15 Das Funktionieren des Phasenverteilers entspricht der im Abfrage-Verwaltungsprozessor PGU gemachten Beschreibung.

Das Signal  $\overline{valid}$  des Busses BUSA ist mit dem Eingang des Differenzierers D40 und dem Freigabeeingang  $\overline{en41}$  eines Decodierers DEC41 verbunden. Die Kippschaltung

20 B41 empfängt auf ihrem Eingang  $\overline{R}$  den Ausgang der Verknüpfung NOU40. Der Ausgang  $\overline{Q}$  der Kippschaltung B41, mit offenem Kollektor, ist mit dem Signal  $\overline{done}$  des Busses BUSA verbunden.

25 Die Signale  $\overline{valid}$  und  $\overline{done}$  sorgen für die Synchronisation des Spionprozessors PE mit den anderen Spionprozessoren des Mehrprozessorsystems: der negative Übergang des Signals  $\overline{valid}$  löst den Differenzierer D40 aus, der einen Impuls erzeugt, der es ermöglicht, den

30 Phasenverteiler zu aktivieren und das Signal  $\overline{done}$  mittels der Kippschaltung B41 in den Null-Zustand zu setzen. Das Ende der Arbeit des Spions wird durch eine Zustandsänderung auf dem Ausgang der Verknüpfung NOU40 signalisiert, diese Zustandsänderung bewirkt mittels der

35 Kippschaltung B41 das Setzen des Signals  $\overline{done}$  in den Logikzustand Eins.

Die Arbeit des Spions ist abhängig von der Art der Anforderung, und zu diesem Zweck wird das Feld Typ des

Busses BUSA an den Eingang des Decodierers DEC41  
 angeschlossen. Die Ausgänge dnp und dmaj des Decodierers  
 DEC41 sind mit den Eingängen einer Verknüpfung OU40  
 verbunden. Die Ausgänge dl, de und dei des Decodierers  
 5 DEC41 sind mit den Eingängen einer Verknüpfung OU41  
 verbunden, wobei die Ausgänge de und dei ebenfalls mit den  
 Eingängen einer Verknüpfung OU42 verbunden sind. Der  
 Ausgang der Verknüpfung OU40 ist mit einem der Eingänge  
 der Verknüpfung NOU40 mittels einer Verknüpfung ET40  
 10 verbunden, die auf ihrem anderen Eingang das Signal  
 theta41 empfängt. Der Ausgang der Verknüpfung OU41 ist mit  
 einem der Eingänge der Verknüpfung ET42 beziehungsweise  
 ET43 verbunden, die auf ihrem anderen Eingang das  
 Phasensignal theta41 beziehungsweise theta44 empfangen.  
 15 Die Ausgänge der Verknüpfungen ET42 und ET43 sind mit dem  
 Eingang  $\bar{S}$  beziehungsweise  $\bar{R}$  einer Kippschaltung B44  
 verbunden. Der Ausgang der Verknüpfung ET42 ist ebenfalls  
 mit den Eingängen  $\bar{S}$  der Kippschaltungen B42 und B43  
 verbunden. Eine Verknüpfung NOU41 empfängt auf ihren  
 20 Eingängen das Phasensignal theta45 und den Ausgang einer  
 Verknüpfung ET45, die auf ihren beiden Eingängen die Phase  
 theta44 und den invertierten Ausgang einer Verknüpfung  
 OU43 empfängt. Der Ausgang der Verknüpfung NOU41 ist mit  
 dem Eingang  $\bar{R}$  der Kippschaltung B42 verbunden. Der Ausgang  
 25  $\bar{Q}$  der Kippschaltung B42 ist mit dem Signal  $\overline{rqst\_PR}$   
 verbunden und das Signal  $\overline{grnt\_PR}$  wird an den Eingang  $\bar{R}$  der  
 Kippschaltung B43 ausgegeben, und zwar an die  
 Freigabeeingänge der Übergabepuffer BV40 und an einen der  
 Eingänge der Verknüpfung OU44. Die Verknüpfung OU44  
 30 empfängt auf ihrem anderen Eingang den Ausgang  $\bar{Q}$  einer  
 Kippschaltung B45, und ihr Ausgang gibt die Übergabepuffer  
 BV41 frei. Der Ausgang der Verknüpfung ET45 ist ebenfalls  
 mit einem der Eingänge der Verknüpfung NOU40 verbunden,  
 die auf einem anderen Eingang das Phasensignal theta45  
 35 empfängt. Der Ausgang der Verknüpfung OU42 ist mit den  
 Eingängen von Verknüpfungen ET46 und ET47 verbunden, die  
 auf ihren Eingängen ebenfalls den Ausgang einer  
 Verknüpfung OU43 und die Phase theta44 beziehungsweise

theta45 empfangen. Der Ausgang  $\bar{Q}$  der Kippschaltung B44  
 gibt das Signal  $\bar{r}_{RG}$  über einen Puffer BV40 aus, der  
 Ausgang  $\bar{Q}$  der Kippschaltung B45 gibt über einen Puffer  
 BV40 das Signal  $\bar{w}_{RG}$  aus. Das Feld Rahmen des gemeinsamen  
 5 Busses BUSA ist über einen Puffer BV40 mit dem Bus  $adr_{RG}$   
 verbunden. Der Bus  $data_{RG}$  ist mit den Ausgängen der  
 Freigabepuffer BV41 und mit dem Eingang des Registers  
 REG40 verbunden, das auf seinem Eingang load40 das  
 Phasensignal theta43 empfängt. Der Ausgang des Registers  
 10 REG40 ist, was den Teil tag betrifft, mit den Eingängen  
 der Puffer BV41 und mit einem der Eingänge eines  
 Komparators COMP40 verbunden. Der Komparator COMP40  
 empfängt auf seinem anderen Eingang den Teil tag des  
 Busses BUSA. Das Freigabebit v aus dem Register REG40, das  
 15 an den Komparator COMP40 angeschlossen ist, hat ihm  
 gegenüber den konstanten Wert 1. Die Bits v, a, m und f am  
 Ausgang des Registers REG40 sind mit einem der Eingänge  
 der Multiplexer MUX40, MUX41, MUX42 beziehungsweise MUX43  
 verbunden. Die Ausgänge dieser Multiplexer liefern den  
 20 Zustand dieser selben Bits an den Eingang der Puffer BV41.  
 Der Multiplexer MUX40 empfängt auf seinen anderen  
 Eingängen die Konstanten Null und Eins, die Eingänge sel40  
 sind mit dem Ausgang einer Verknüpfung ET48 und mit einem  
 Signal  $d_{lle}$  verbunden. Der Multiplexer MUX41 empfängt auf  
 25 seinem anderen Eingang die Konstante Eins, die ausgewählt  
 wird, wenn sich sein Eingang sel41, der ein Signal  $d_{lle}$   
 aus dem Verwaltungsprozessor PGP empfängt, im Logikzustand  
 Eins befindet. Der Multiplexer MUX42 empfängt auf seinen  
 anderen Eingängen die Konstante Eins und das Signal  
 30  $r_{adrbloc\_SR}$ , seine Eingänge sel42 empfangen die Signale  
 $d_{lei}$  und  $d_{lle}$  aus dem Verwaltungsprozessor PGP. Der  
 Multiplexer MUX43 empfängt auf seinem anderen Eingang die  
 Konstante Eins, die ausgewählt wird, wenn sich sein  
 Eingang sel43, der mit dem Ausgang einer Verknüpfung ET49  
 35 verbunden ist, im Logikzustand Eins befindet. Die  
 Verknüpfung ET49 empfängt auf ihren Eingängen den Ausgang  
 eg40 des Komparators COMP40, das invertierte Signal f und  
 das Signal m. Die Verknüpfung ET48 empfängt auf ihren

Eingängen den Ausgang eg40, das invertierte Signal m, das Signal dlei und den Ausgang der Verknüpfung OU42. Die Verknüpfung OU43 empfängt auf einem ihrer Eingänge das Signal eg40 und auf ihrem anderen Eingang das Signal dlle.

5 Der Ausgang der Verknüpfung ET49 ist ebenfalls mit dem Eingang load41 der Warteschlange FIFO verbunden, die bereits in dem Verwaltungsprozessor des Seriendrahts PGS beschrieben ist. Die Felder Rahmen und tag des Busses BUSA sind mit dem Eingang der Warteschlange FIFO verbunden. Die

10 Signale dlei, dlle und r\_adrbloc\_SP kommen vom Verwaltungsprozessor des parallelen Drahts PGP, der im übrigen das Signal dei des Decodierers DEC41 empfängt.

Das Ganze funktioniert folgendermaßen: die Aktivierung des Signals valid initialisiert den

15 Phasenverteiler DP\_E und gibt den Decodierer DEC41 frei, der die Aktivierung eines Ausganges bewirkt, und zwar abhängig von der Information "Typ", die die Art der sich im Gang befindlichen Anforderung an den gemeinsamen Bus BUSA codiert. Der aktive Ausgang kann sein:

- 20 - dnp: Lese-Abfrage von nicht geteilten Daten.  
Das Signal done wird bei Erscheinen der Phase theta41 inaktiviert;
- dmaj: Schreib-Abfrage für ein Block-Update.  
Das Signal done wird bei der Phase theta41 inaktiviert;
- 25 - dl: Block-Lese-Abfrage;  
- de: Block-Schreib-Abfrage;  
- dei: Informations-Schreib-Abfrage.

Diese drei Fälle benötigen einen Lese-Zugriff auf das Verzeichnis RG, und die beiden letzteren ein eventuelles erneutes Schreiben des Verzeichnisses. Dazu

30 wird von der Kippschaltung B42 (Signal rqst\_PR) eine Zugriffs-Abfrage auf den Verzeichnis-Verwaltungsprozessor PGR gesandt, wobei die Kippschaltung B43 den Phasenverteiler bis zur Zugriffserlaubnis unterdrückt,

35 die durch das Signal grnt\_PR gegeben wird. Das Lesen erfolgt somit von theta41 bis theta44 (Kippschaltung B44) und das eventuelle Schreiben von theta44 bis theta45 (Kippschaltung B45) mit Einspeichern des Stichworts

während der Phase  $\theta_{43}$  in das Register REG40. Ist der Block im Cache-Speicher MC nicht vorhanden ( $eg_{40} = 0$ ), dann wird bei der Phase  $\theta_{44}$  das Signal  $\overline{done}$  inaktiviert. Ist der Block im Cache-Speicher vorhanden

5 ( $eg_{40} = 1$ ), dann:

- ist  $m = 1$ , wird eine Anforderung nach vollständiger Entladung aktiviert (Aktivierung der Verknüpfung ET49), und zwar unter der Bedingung, daß sich dieser Block nicht schon in der Warteschlange ( $f = 0$ ) befindet: das einzige modifizierte Bit ist  $f$ , das während des erneuten Schreibens des Stichworts auf Eins gesetzt wird,

10

- ist  $m = 0$ , wird der Block durch den Multiplexer MUX40 (Aktivierung der Verknüpfung ET48)

15 ungültig gemacht,

- ist die Anforderung lokal ( $\overline{dlle}$  oder  $\overline{dle}$  aktiv) dann:

1) werden beim Lesen oder Schreiben die Bits  $v$  und  $a$  auf 1 und  $m$  auf 0 (Lesen) oder auf 1 (Schreiben) (Zustand des Signals  $\overline{f\_adrbloc\_SP}$ ) gesetzt,

20

2) wird beim Informations-Schreibvorgang das Bit  $m$  auf 1 gesetzt.

In diesen letzteren Fällen, die ein Neuschreiben in das Verwaltungsverzeichnis RG erfordern, wird das Signal  $\overline{done}$  bei der Phase  $\theta_{45}$  inaktiviert.

25

Der Verwaltungsprozessor des parallelen Drahts PGP, von dem ein Beispiel in Abbildung 16 dargestellt ist, hat die Aufgabe, den gemeinsamen Bus BUSA abzufragen und die angeforderte Transaktion durchzuführen, entweder durch den Abfrage-Verwaltungsprozessor PGU oder durch den Verwaltungsprozessor der Serienverbindung PGS.

30

Bei einer Anforderung aus dem Abfrage-Verwaltungsprozessor PGU kann es sich nur um eine Informations-Schreib-Abfrage handeln. Eine Anforderung aus dem Verwaltungsprozessor der Serienverbindung PGS ist entweder eine Block-Lese- oder Block-Schreib-Anforderung oder aber eine Block-Update-Anforderung.

35

Der Verwaltungsprozessor des parallelen Drahts PGP umfaßt eine Kippschaltung B60, die durch ihren Dateneingang D mit dem Signal  $\overline{rqst\_UP}$  verbunden ist. Eine Kippschaltung B61 ist durch ihren Dateneingang D mit dem Signal  $\overline{rqst\_SP}$  verbunden. Die Ausgänge Q und  $\overline{Q}$  einer Kippschaltung B62 sind mit dem Taktgebereingang der Kippschaltung B60 beziehungsweise B61 verbunden. Der Ausgang  $\overline{Q}$  der Kippschaltung B62 ist auf deren Dateneingang zur Schleife geschaltet. Die Ausgänge  $\overline{Q}$  der Kippschaltungen B60 und B61 sind mit den Eingängen einer Verknüpfung OU60 verbunden. Der Ausgang der Verknüpfung OU60 ist einerseits mit einem Differenzierer D60 verbunden und andererseits, und zwar invertiert, mit einem Eingang einer Verknüpfung ET60, die auf ihrem anderen Eingang das Signal des allgemeinen Taktgebers h empfängt. Der Ausgang der Verknüpfung ET60 ist mit dem Taktgebereingang der Kippschaltung B62 verbunden. Der Ausgang des Differenzierers D60 ist mit dem Eingang  $\overline{S}$  einer Kippschaltung B63 verbunden. Der Ausgang  $\overline{Q}$  der Kippschaltung B63 gibt ein Signal  $\overline{rqsti}$  zum Arbiter AB aus, und ihr Eingang  $\overline{R}$  ist mit dem Ausgang einer Verknüpfung ET62 verbunden. Das aus dem Arbiter AB stammende Signal  $\overline{grnti}$  ist mit den Verknüpfungen OU62 und NOU60 verbunden. Die Verknüpfung OU62 empfängt auf ihrem anderen Eingang das invertierte Signal  $\overline{valid}$ , und ihr Ausgang ist mit dem Eingang eines Differenzierers D61 verbunden. Der Ausgang dieses Differenzierers D61 ist mit einem der Eingänge der Verknüpfung ET62 und dem Eingang  $\overline{S}$  einer Kippschaltung B64 verbunden. Der Ausgang Q dieser Kippschaltung B64 ist mit dem Freigabeeingang von Übergabepuffern BV60 und nach Negation mittels eines Inverters mit offenem Kollektor I60 mit dem Signal  $\overline{valid}$  verbunden. Das Signal  $\overline{done}$  ist mit dem Eingang eines Differenzierers D62 verbunden. Der Ausgang dieses Differenzierers D62 ist mit dem Eingang  $\overline{R}$  der Kippschaltung B64 und mit einem der Eingänge der Verknüpfung NOU60 verbunden. Der Ausgang dieser Verknüpfung NOU60 ist mit einem der Eingänge von

Verknüpfungen NET60 und NET61 verbunden, die auf ihrem anderen Eingang den Ausgang  $\overline{Q}$  der Kippschaltung B60 beziehungsweise B61 empfangen. Die Ausgänge  $\overline{Q}$  der Kippschaltungen B60 und B61 sind ebenfalls mit den  
 5 Freigabeeingängen des Übergabepuffers BV61 beziehungsweise BV62 verbunden. Der Ausgang der Verknüpfung NET60 ist einerseits mit dem Eingang  $\overline{S}$  der Kippschaltung B60 verbunden, andererseits mit einem der Eingänge einer Verknüpfung ET63, die auf ihrem anderen Eingang den  
 10 Ausgang eines Differenzierers D63 empfängt. Der Ausgang der Verknüpfung ET63 gibt das Signal  $\overline{acq\_UP}$  zum Verwaltungsprozessor PGU aus. Der Ausgang der Verknüpfung NET61 ist mit dem Eingang  $\overline{S}$  der Kippschaltung B61 verbunden und liefert das Signal  $\overline{ack\_SP}$ . Der Bus  
 15  $adr\_bloc\_UP$  ist mit dem Eingang der Freigabepuffer BV61 und mit einem der Eingänge eines Komparators COMP60 verbunden.

Der Bus  $adr\_bloc\_SP$  ist mit dem Eingang der Freigabepuffer BV62 verbunden. Die Pufferausgänge BV61 und  
 20 BV62 sind miteinander und mit dem Eingang der Freigabepuffer BV60 verbunden. Der Ausgang der Puffer BV60 ist mit dem gemeinsamen Bus BUSA verbunden. Die Verknüpfungen OU63 und OU64 empfangen auf ihren Eingängen den Ausgang  $\overline{Q}$  der Kippschaltung B60 beziehungsweise B61,  
 25 das Logiksignal  $\overline{grnti}$  und das Signal  $maj$  für OU64. Der Ausgang der Verknüpfung OU63 gibt das Signal  $\overline{dlel}$  aus, und der Ausgang der Verknüpfung OU64 das Signal  $dlle$ . Der andere Eingang des Komparators COMP60 empfängt die Felder tag und Rahmen des gemeinsamen Busses BUSA. Der Eingang  
 30  $en60$  des Komparators COMP60 ist mit dem Ausgang der Verknüpfung ET61 verbunden, die auf ihren Eingängen das invertierte Signal  $\overline{grnti}$ , das Signal  $dei$  und das invertierte Signal  $\overline{rqst\_UP}$  empfängt. Der Ausgang  $eg60$  des Komparators COMP60 ist mit dem Eingang des Differenzierers  
 35 D63 verbunden. Der Ausgang dieses Differenzierers ist ebenfalls mit den Eingängen  $\overline{R}$  der Kippschaltungen B60 und B61 und mit dem anderen Eingang der Verknüpfung ET62 verbunden.

Das Ganze funktioniert folgendermaßen:

Die den Verknüpfungen ET60 und OU60 zugeordneten Kippschaltungen B60, B61 und B62 bilden einen lokalen Arbitr. Dieser Arbitr untersucht abwechselnd die  
 5 Abfragen  $\overline{rqst\_UP}$  und  $\overline{rqst\_SP}$  und leitet sie durch das Signal  $\overline{rqsti}$  zum Arbitr AB des gemeinsamen Busses BUSA weiter. Die Zugriffserlaubnis wird durch die Freigabe des Signals  $\overline{grnti}$  gegeben und der Buszyklus läuft ab, sobald das Signal  $\overline{valid}$ , das den Arbitr AB freimacht,  
 10 inaktiviert worden ist. Die Aktivierung des Signals  $\overline{done}$  macht den lokalen Arbitr frei: die Transaktion ist beendet.

Kommt die Abfrage vom Verwaltungsprozessor der Serienverbindung PGS, dann zeigen die Signale  $\overline{dle1}$  und  
 15  $\overline{dle}$  dem zugeordneten Spionprozessor PE die Art des Updates der Blockzustandsbits an, das in dem Verzeichnis RG durchzuführen ist.

Kommt die Abfrage vom Abfrage-Verwaltungsprozessor PGU her, dann findet bei Entdecken  
 20 eines Informations-Schreibvorgangs auf demselben Block (Signal  $\overline{dei}$  aus dem Spionprozessor PE), ein sofortiges Freimachen statt: der Abfrage-Verwaltungsprozessor PGU leitet nach Konsultation des Verzeichnisses (der Block ist ungültig gemacht worden) seine Anforderung zum  
 25 Verwaltungsprozessor der Serienverbindung PGS.

Der Speicher-Verwaltungsprozessor PGM, von dem ein Beispiel in Abbildung 17 dargestellt ist, hat die Aufgabe, für das Lesen und Schreiben eines Blocks im Hauptspeicher RAM zu sorgen und an der Aufrechterhaltung  
 30 der Kohärenz der Information in den verschiedenen Cache-Speichern MC des Mehrprozessorsystems teilzunehmen.

Zu diesem Zweck umfaßt er einen Differenzierer D80, der auf seinem Eingang ein Signal  $\overline{valid}$  empfängt und dessen Ausgang mit den Eingängen  $\overline{S}$  von Kippschaltungen B80  
 35 und B81 sowie mit dem Eingang  $\overline{clr\_80}$  eines Schieberegisters SR80 verbunden ist. Auf dem Ausgang  $\overline{Q}$  der Kippschaltung B80, mit offenem Kollektor, wird das Signal  $\overline{done}$  ausgegeben. Der Ausgang Q der Kippschaltung B81 ist

mit dem Eingang serial\_in80 des Registers SR80 verbunden; diese Kippschaltung B81 ist durch ihren Eingang  $\bar{R}$  an den invertierten Ausgang theta81 des Registers SR80 verbunden. Das Register SR80 empfängt auf seinem Eingang clk80 das

5 Signal des allgemeinen Taktgebers h, und sein Freigabeeingang en80 ist immer aktiv. Die Kippschaltung B81 und das Schieberegister SR80 bilden einen Phasenverteiler DP\_M. Das Signal  $\overline{\text{valid}}$  wird ebenfalls an den Freigabeeingang en81 eines Decodierers DEC80

10 ausgegeben. Dieser Decodierer ist durch seinen Dateneingang mit dem Teil "Typ" des gemeinsamen Busses BUSA verbunden und liefert die Signale dnp, dl, de, ei und maj. Ein Schreib-Lese-Speicher RAMFG mit einer Breite von 2 Bits (ro beziehungsweise rw genannt) empfängt auf seinem

15 Adreßbus die Felder tag und Rahmen des gemeinsamen Busses BUSA. Der Datenbus dieses Speichers, aus den Bits ro und rw gebildet, ist einerseits mit einer Logik PAL80 verbunden, andererseits mit einer Verknüpfung ET80, und zwar direkt für rw und invertiert für ro. Die Logik PAL80

20 ist mit dem Feld Typ verbunden und empfängt die Logiksignale cl, r/w, s/n, mff und en82: das Signal cl kommt aus einer Kippschaltung B82, die Signale r/w und s/n aus einer assoziativen Warteschlange AFIFO, das Signal mff aus einer Verknüpfung ET81 und das Signal en82 aus einer

25 Kippschaltung B83, die auf ihren Eingängen  $\bar{S}$  und  $\bar{R}$  invertierte Signale theta82 beziehungsweise theta81 aus dem Phasenverteiler DP\_M empfängt. Die Logik PAL verdrahtet auf ihren Ausgängen ro - rw die folgenden logischen Gleichungen: dnp = 10; dl.mff = 10; dl.mff = 01;

30 de = 01; maj.cl = 00; ei = 01; maj.cl.s/n.r/w = 10; maj.cl.s/n.r/w = 01. Der Ausgang der Verknüpfung ET80 ist mit dem Eingang D einer Kippschaltung B84 verbunden, die auf ihrem Taktgebereingang die Phase theta82 empfängt. Der Ausgang Q dieser Kippschaltung ist mit einem der Eingänge

35 der Verknüpfung ET81 verbunden, die auf ihrem anderen Eingang den Ausgang der Verknüpfung OU80 empfängt. Die beiden Eingänge dieser Verknüpfung OU80 sind mit den Ausgängen de und dl des Decodierers DEC80 verbunden. Der

Leseeingang  $r$  des Speichers RAMFG ist mit der Phase  $\theta_{81}$  und der Schreibeingang  $w$  mit dem Ausgang einer Verknüpfung ET82 verbunden. Die Verknüpfung ET82 empfängt auf ihren Eingängen das Signal  $\theta_{83}$  und den Ausgang einer Verknüpfung ET83, deren Eingänge mit dem Signal  $s/\bar{n}$  und der Phase  $\theta_{87}$  verbunden sind. Der Ausgang  $dnp$  des Decodierers DEC80 ist mit den Verknüpfungen ET84 und ET85 verbunden, die auf ihrem anderen Eingang die Phase  $\theta_{81}$  beziehungsweise  $\theta_{85}$  empfangen. Das Signal  $s/\bar{n}$  wird ebenfalls an Verknüpfungen ET86 und ET87 ausgegeben, die auf ihrem anderen Eingang die Phase  $\theta_{86}$  beziehungsweise  $\theta_{90}$  empfangen. Der Ausgang  $mff$  der Verknüpfung ET81 ist ebenfalls mit einer Verknüpfung ET88 verbunden, die auf ihrem anderen Eingang die Phase  $\theta_{83}$  empfängt, und nach Negation mit Verknüpfungen ET89 und ET90 verbunden, die auf ihrem anderen Eingang die Phase  $\theta_{83}$  beziehungsweise  $\theta_{87}$  empfangen. Der Ausgang der Verknüpfung ET88 ist mit dem Eingang  $wff$  der Warteschlange AFIFO verbunden. Die Ausgänge der Verknüpfungen ET84, ET86 und ET89 sind mit den Eingängen einer Verknüpfung OU81 verbunden, deren Ausgang mit dem Eingang  $S$  einer Kippschaltung B85 verbunden ist. Die Ausgänge der Verknüpfungen ET85, ET87 und ET90 sind mit den Eingängen einer Verknüpfung OU82 verbunden, deren Ausgang mit dem Eingang  $R$  der Kippschaltung B85 verbunden ist. Das Signal  $s/\bar{n}$  ist ebenfalls, invertiert, mit einer Verknüpfung ET91 verbunden, die auf ihrem anderen Eingang die Phase  $\theta_{89}$  empfängt. Der Ausgang der Verknüpfung ET91 ist mit dem Eingang einer Verknüpfung NOU80 verbunden, die auf ihrem anderen Eingang den Ausgang der Verknüpfung OU82 empfängt. Der Ausgang der Verknüpfung NOU80 ist mit dem Eingang  $\bar{R}$  der Kippschaltung B80 verbunden. Der Ausgang  $maj$  des Decodierers DEC80 ist mit dem Eingang von Verknüpfungen ET92, ET93, ET94 und ET95 verbunden, die auf ihrem anderen Eingang die Phase  $\theta_{81}$ ,  $\theta_{85}$ ,  $\theta_{85}$  beziehungsweise  $\theta_{91}$  empfangen. Die invertierten Ausgänge der Verknüpfungen ET92 und ET93 sind mit dem Eingang  $S$  beziehungsweise  $R$  einer Kippschaltung B86

verbunden und die der Verknüpfungen ET94 und ET95 mit den Eingängen S und R einer Kippschaltung B82. Der Ausgang Q der Kippschaltung B82 erzeugt das Logiksignal cl, das ebenfalls an den Eingang cff der Warteschlange AFIFO und  
 5 an den Steuereingang sel80 eines Multiplexers MUX80 ausgegeben wird. Der Teil tag, Rahmen des gemeinsamen Busses BUSA ist mit dem Dateneingang der Warteschlange AFIFO und mit einem der Dateneingänge des Multiplexers MUX80 verbunden. Der Ausgang dl des Decodierers DEC80 ist  
 10 ebenfalls mit einem der Dateneingänge der Warteschlange AFIFO verbunden, um das Lese-/Schreib-Signal l/e zu erzeugen. Der Datenausgang der Warteschlange AFIFO ist mit dem anderen Eingang des Multiplexers MUX80 verbunden. Der Ausgang des Multiplexers MUX80 ist, was den Teil  
 15 "tag.Rahmen" betrifft, mit dem Adreßbus des Hauptspeichers RAM verbunden und, was den Teil "Feld cpu" betrifft, mit den Eingängen von Decodierern DEC81 und DEC82. Der Ausgang  $\overline{Q}$  der Kippschaltung B86 ist mit dem Schreibeingang des Hauptspeichers RAM und dem Eingang en84 des Decodierers  
 20 DEC82 verbunden. Der Ausgang Q der Kippschaltung B85, leicht verzögert, ist mit dem Leseingang des Hauptspeichers RAM und mit einem der Eingänge einer Verknüpfung ET96 verbunden, die auf ihrem anderen Eingang den Ausgang der Verknüpfung OU82 empfängt. Der Ausgang der  
 25 Verknüpfung ET96 ist mit dem Eingang en83 des Decodierers DEC81 verbunden. Der Ausgang j des Decodierers DEC81 ist mit dem Freigabeeingang der Übergabepuffer des Speicher-Schieberegisters RDM<sub>j</sub> verbunden und der Ausgang j des Decodierers DEC82 mit dem Ladeeingang des besagten  
 30 Speicher-Schieberegisters RDM<sub>j</sub>.

Das Ganze funktioniert folgendermaßen:

Die Aktivierung des Signals valid bewirkt die Auslösung des Phasenverteilers DP\_M und die Freigabe des Decodierers DEC80, die es ermöglichen wird, die Art der  
 35 Abfrage zu bestimmen. Die Phase theta81 wird benutzt, um den Zustand der Bits, die dem angeforderten Block entsprechen, im Speicher RAMFG zu lesen, und die Kombination ro.rw wird in der Kippschaltung B84

gespeichert. Ein erster Schreibvorgang findet in dem Speicher RAMFG bei der Phase  $\theta_{83}$  statt, was es ermöglicht, die Zustandsbits zu aktualisieren. Ihr Wert wird von der Logik PAL80 geliefert und ermöglicht es, folgende Verkettungen zu erhalten:

- bei einer Block-Abfrage von nichtgeteilten Daten (dnp), unabhängig vom Zustand der Bits ro.rw (rw ist zwangsweise auf Null), wird der Zustand 10 gesetzt ("Block für das Lesen ausgesandt");

- bei einer Block-Lese-Abfrage (dl) oder Block-Schreib-Abfrage (de) wird, wenn ro.rw = 01 ist, die Abfrage bei der Phase  $\theta_{83}$  in die Warteschlange gestellt und der Zustand auf 01 gesetzt (dies ist in der Tat der zuvor beschriebene Zustand), ansonsten wird im Falle eines Lesevorgangs der Zustand auf 10 gesetzt ("Block zum Lesen ausgesandt") und im Fall des Schreibvorgangs der Zustand auf 01 gesetzt ("Block zum Schreiben ausgesandt");

- im Falle einer Update-Abfrage (maj), wird der Zustand auf 00 gesetzt ("Block nicht ausgesandt"). In diesen verschiedenen Fällen wird das Lesen oder das Schreiben in den Hauptspeicher RAM zum oder vom Speicher-Schieberegister RDM<sub>j</sub> vorgenommen, das durch das Feld "cpu" des gemeinsamen Busses BUSA identifiziert worden ist. In dem ausgewählten Beispiel beträgt die Zyklusdauer des Speichers RAM 4 Perioden des allgemeinen Taktgebers h. Beim Lesen nicht geteilter Daten, läuft der Zyklus von  $\theta_{81}$  bis  $\theta_{85}$  ab, in den anderen Fällen von  $\theta_{83}$  bis  $\theta_{87}$ . Das Schreiben erfolgt von  $\theta_{81}$  bis  $\theta_{85}$ ;

- beim Informations-Schreibvorgang, bewirkt dieses keine Datenbewegung, die Zustandsbits werden jedoch auf den Wert 01 gesetzt (der Ausgangszustand ist in diesem Fall zwangsläufig 10);

- bei einer Update-Anforderung, wird systematisch eine Konsultation der Warteschlange AFIFO durchgeführt. Diese Konsultation kann das Lesen eines Blocks bewirken, und zwar dann, wenn sich eine

Zentraleinheit CPU für ein Update dieses Blocks in der Warteschlange AFIFO befindet.

Das Lesen erfolgt von theta86 bis theta90 und der Zustand der Bits wird auf 10 gesetzt (Lese-  
 5 Anforderung) oder auf 01 (Schreib-Anforderung). Jede Operation wird durch das Zurücksetzen der Kippschaltung B80 auf Null beendet, die das Signal done aktiviert. Diese Inaktivierung kann bei der Phase theta85, theta87 oder theta91 erfolgen, je nach der angeforderten  
 10 Operation, oder bei theta89, falls die Konsultation der Warteschlange ein negatives Ergebnis bringt.

Die assoziative Warteschlange ist nicht im Detail beschrieben. Sie besteht nach herkömmlicher Art und Weise aus einem assoziativen Speicher, der als  
 15 Warteschlange benutzt wird. Die Anzahl der Wörter dieses Speichers ist gleich der Anzahl der Zentraleinheiten CPU des Mehrprozessorsystems. Ein interner "daisy-chain" identifiziert auf jeder Phase theta81 das nächste zum Schreiben bestimmte Wort, was gegebenenfalls bei der Phase  
 20 theta83 durch das Signal wff erfolgt. Das Signal cff löst ab der Phase theta85 einen Vergleich aus, nachdem die Kippschaltungen des antwortenden Speichers bei der Phase theta84 auf Null zurückgesetzt worden sind. Das Ergebnis des Vergleichs wird an das Signal s/n (some/none)  
 25 weitergegeben, und der Inhalt des betreffenden Worts ist auf dem Datenausgang ab der Phase theta86 verfügbar. Dieses Wort wird dann bei der Phase theta90 ungültig gemacht.

In der oben beschriebenen Architektur werden  
 30 die Spionprozessoren PE<sub>j</sub> bei jedem Adressentransfer auf dem gemeinsamen Bus BUSA beansprucht, mit eventueller Konsultation ihres Cache-Speichers MC<sub>j</sub>. Diese Konsultation ist in den meisten Fällen überflüssig (geringe Wahrscheinlichkeit des Vorhandenseins der dem Transfer  
 35 entsprechenden Blockadresse in den Cache-Speichern).

Es muß angemerkt werden, daß der Speicher-Verwaltungsprozessor PGM Zustandsbits der Blöcke aufrechterhält und eine zentralisierte Verwaltung der

Aufrechterhaltung der Kohärenz möglich macht. Zu diesem Zweck kann zu der oben beschriebenen Architektur (Abbildung 10) ein paralleler Synchronisationsbus hinzugefügt werden, der nach demselben Algorithmus funktioniert, wie der Synchronisationsbus SYNCHRO der nachstehend beschriebenen Variante. Die Spionprozessoren sind genau genommen nun keine Spione mehr (da sie an den Synchronisationsbus und nicht an den gemeinsamen Bus BUSA angeschlossen sind) und werden als Prozessoren zur Aufrechterhaltung der Kohärenz (PMC<sub>j</sub> bei der Variante der Abbildung 18) bezeichnet. So bleibt der Speicher-Verwaltungsprozessor PGM bei jedem Transfer auf dem gemeinsamen Bus BUSA beansprucht, jedoch werden die Prozessoren zur Aufrechterhaltung der Kohärenz von dem Prozessor PGM nur beansprucht, wenn sie von dem Transfer betroffen sind.

Abbildung 18 stellt eine Variante dar, in der die Kohärenz nach dem oben beschriebenen Prinzip aufrechterhalten wird. Diese Variante nimmt die allgemeine Architektur der Abbildung 6 wieder auf, mit Blockadressen, die mittels der Serienverbindungen LS<sub>j</sub> übertragen werden. Dieses System umfaßt einen parallelen Synchronisationsbus SYNCHRO, der dieselbe logische Struktur aufweist, wie der gemeinsame Bus BUSA, der jedoch allein vom Speicher-Verwaltungsprozessor PGM gesteuert wird.

Die Struktur der Zentraleinheit UC<sub>j</sub> entspricht der in Abbildung 10 dargestellten, mit einigen Änderungen:

- die Struktur des Cache-Speichers MC<sub>j</sub> sowie die Struktur des Verwaltungsverzeichnisses RG<sub>j</sub> bleibt dieselbe,
- der parallele Verwaltungsprozessor PGP<sub>j</sub> verschwindet, da es keinen gemeinsamen Bus BUSA mehr gibt, und die Funktionen, die ihm zugeordnet waren, werden dem Verwaltungsprozessor der Serienverbindung PGS<sub>j</sub> zugeordnet;
- der Spionprozessor PE<sub>j</sub> wird durch einen Prozessor zur Aufrechterhaltung der Kohärenz PMC<sub>j</sub> ersetzt, der mit der Aufrechterhaltung der Zustandsbits der Blöcke in dem Cache-Speicher MC<sub>j</sub> befaßt ist, um für die Kohärenz

zu sorgen, und der allein vom Speicher-Verwaltungsprozessor PGM<sub>j</sub> über den Synchronisationsbus SYNCHRO aktiviert wird;

- der Abfrage-Verwaltungsprozessor PGU hat nur noch einen einzigen Partner: den Verwaltungsprozessor der Serienverbindung PGS<sub>j</sub>, zu dem er alle seine Abfragen leitet;

- der Verwaltungsprozessor der Serienverbindung PGS<sub>j</sub> hat die Aufgabe, für den Adressen- und Datentransfer zu sorgen, entsprechend dem System, dessen Prinzip bezüglich der Abbildung 6 beschrieben ist, wobei jede Adresse durch die Art der Abfrage ein Präfix erhält;

- die Funktionalitäten des Speicher-Verwaltungsprozessors PGM sind jene, die bezüglich Abbildung 17 beschrieben sind, seine Aktivierung erfolgt nicht mehr durch das Signal valid, das verschwindet (da es zuvor dem gemeinsamen Bus BUSA zugeordnet war), sondern durch den Arbiter ABM, der in dem System der Abbildung 6 beschrieben ist und der die Dienstanforderungen, die durch die Serienverbindungen übertragen werden, parallel-seriell umsetzt. Der Speicher RAMFG besteht ebenfalls aus einem zusätzlichen Feld cpu, das den Zustandsbits ro.rw zugeordnet ist.

- Das in Abbildung 18 dargestellte Durchführungsbeispiel funktioniert wie folgt:

- Jede Abfrage des Verarbeitungsprozessors CPU aktiviert den Abfrage-Verwaltungsprozessor PGU<sub>j</sub> mit der Angabe Lesen oder Schreiben und Code oder Daten. Dieser Prozessor fordert bei dem Verzeichnis-Verwaltungsprozessor PGR<sub>j</sub> einen Zugriff zum Verzeichnissesverzeichnis RG<sub>j</sub> an. Die Konsultation des Verzeichnisses führt zu einem der folgenden Fälle:

- der Block ist im Cache-Speicher MC<sub>j</sub> im "nicht modifiziert"-Zustand (m = 0) vorhanden; ist die Anforderung eine Lese-Anforderung, dann wird die angeforderte Information aus dem Cache-Speicher MC<sub>j</sub> geholt und dem Verarbeitungs-Prozessor CPU<sub>j</sub> geliefert. Ist die

Anforderung eine Schreib-Anforderung, dann wird eine Informations-Schreib-Abfrage ei an den Verwaltungsprozessor der Serienverbindung PGS<sub>j</sub> gesandt;

- der Block ist im Cache-Speicher MC<sub>j</sub> im "modifiziert"-Zustand (m = 1) vorhanden; der Anforderung, Lesen oder Schreiben, wird entsprochen;

- der Block ist im Cache-Speicher MC<sub>j</sub> nicht vorhanden; eine Blockabfrage Lesen oder Schreiben wird an den Verwaltungsprozessor der Serienverbindung PGS<sub>j</sub> übertragen.

So können die an den Verwaltungsprozessor der Serienverbindung gestellten Abfragen sein: eine Lese-Anforderung nicht geteilter Daten (Code): dnp, eine Lese-Anforderung des Blocks: dl, eine Lese-Anforderung des Blocks, um dort zu schreiben: de, eine Informations-Schreib-Anforderung: ei.

Zu diesen verschiedenen Zuständen muß der Update-Zustand maj, der dem vollständigen Entladen eines Blocks entspricht, hinzugefügt werden, entweder auf Anforderung des Prozessors zur Aufrechterhaltung der Kohärenz PMC<sub>j</sub> oder um eine Blockstelle im Cache-Speicher freizumachen. Die so mit einem Präfix versehenen Adressen werden über die Serienverbindungen LS<sub>j</sub> übertragen und beansprucht, gemäß dem bei der Beschreibung der Architektur der Abbildung 6 angesprochenen Prinzip, den Arbiter ABM, wenn:

- bei einem Block-Lesen die Adresse übertragen wird,

- bei einem Block-Schreiben die Adresse und die Daten übertragen werden.

Diese Abfragen werden von dem Speicher-Verwaltungsprozessor PGM sequentiell verarbeitet, wobei dieser Prozessor dieselbe allgemeine Struktur aufweist, wie die bezüglich der Abbildung 17 beschriebene. Die Verarbeitung erfolgt wie folgt:

1/ dnp: Anforderung nach nicht geteilten Daten. Der Block wird übertragen und nimmt den Zustand ro.rw = 10 an.

2/ dl: Lese-Anforderung eines Blocks.

Befindet sich der Block in dem Zustand "nicht ausgesandt" (ro.rw = 00) oder "zum Lesen ausgesandt" (ro.rw = 10), wird er übertragen und nimmt den Zustand  
 5 ro.rw = 01 an oder behält ihn.

Befindet sich der Block in dem Zustand "zum Schreiben ausgesandt", wird die Anforderung in die Warteschlange AFIFO gestellt. Der Speicher-Verwaltungsprozessor PGM findet dann in dem Feld cpu des  
 10 Speichers RAMFG die Adresse des Cache-Speichers  $MC_i$ , die die aktualisierte Version des angeforderten Blocks enthält. Eine Abfrage für das vollständige Entladen wird dann auf dem Synchronisationsbus SYNCHRO nur an den Prozessor zur Aufrechterhaltung der Kohärenz  $PMC_i$ , der dem  
 15 betreffenden Cache-Speicher  $MC_i$  zugeordnet ist, ausgesandt. Diese Anforderung kann als adressierter Befehl bezeichnet werden.

Man kann feststellen, daß der Prozessor zur Aufrechterhaltung der Kohärenz  $PMC_i$  das zugeordnete  
 20 Verzeichnis  $RG_i$  nicht zu konsultieren braucht, da der Speicher-Verwaltungsprozessor PGM weiß, daß er der einzige Besitzer der aktualisierten Kopie ist. Seine Rolle besteht lediglich darin, die Abfrage zu entnehmen und sie in der zugeordneten Warteschlange  $FIFO_i$  abzulegen.

25 3/ de: Lese-Anforderung eines Blocks, um darin zu schreiben.

Befindet sich der Block in dem Zustand "nicht ausgesandt" (ro.rw = 00), wird er übertragen und nimmt den Zustand "zum Schreiben ausgesandt" (ro.rw = 01) an.

30 Befindet sich der Block in dem Zustand "zum Lesen ausgesandt" (ro.rw = 10), dann sendet der Speicher-Verwaltungsprozessor einen allgemeinen Befehl für das Ungültigmachen des Blocks und überträgt dann den Block in dem Zustand "zum Schreiben ausgesandt" (ro.rw = 01). Der  
 35 allgemeine Befehl bewirkt die Aktivierung aller Prozessoren zur Aufrechterhaltung der Kohärenz  $PMC_j$ , die ganz genau dieselben Operationen ausführen, wie die, die für das System der Abbildung 10 beschrieben sind.

Befindet sich der Block in dem Zustand "zum Schreiben ausgesandt", dann wird die Anforderung in die Warteschlange AFIFO gestellt. Wie zuvor sendet der Speicher-Verwaltungsprozessor PGM einen Befehl, der nur  
 5 für den Besitzer der aktualisierten Kopie bestimmt ist.

4/ maj: Schreib-Anforderung eines Blocks nach einem vollständigen Entladen.

Der Funktionsalgorithmus ist in diesem Fall genau derselbe, wie der bezüglich Abbildung 17 für den  
 10 Prozessor PGM beschriebene.

Es muß angemerkt werden, daß das Schreibquittier-Problem in dieser Durchführungsart ganz von selbst durch einen adressierten Quittierbefehl gelöst wird.

15 5/ ei: Informations-Schreiben.

Dieser Fall wird direkt auf dem gemeinsamen Bus BUSA in der in Abbildung 10 dargestellten Architektur verarbeitet. In der hier angestrebten Durchführungsart und um die Synchronisation zu garantieren, wird diese  
 20 Operation von dem Speicher-Verwaltungsprozessor PGM durchgeführt.

Befindet sich der Block in dem Zustand "zum Lesen ausgesandt", dann wird ein Befehl, der gleichzeitig allgemein und adressiert ist, gesandt: adressiert  
 25 insofern, als daß der betreffende Prozessor zur Aufrechterhaltung der Kohärenz  $PMC_j$  das Quittieren der Anforderung nach dem Informations-Schreibvorgang registriert und den betreffenden Block im "modifiziert"-Zustand im Verzeichnis  $RG_j$  umsetzt, allgemein  
 30 insofern, als daß alle anderen Prozessoren  $PMC_i$  diesen Block in ihrem Verzeichnis ungültig machen müssen.

Der Block in dem Zustand "zum Schreiben ausgesandt" zeigt an, daß während der Wartezeit der Abfrageverarbeitung eine Informations-Schreib-Anforderung  
 35 auf diesem selben Block verarbeitet worden ist. In diesem Fall wird die Informations-Schreib-Anforderung in eine Schreib-Anforderung de umgewandelt, und es folgt dieselbe

Verarbeitung, wie in dem dem Schreibvorgang entsprechenden Fall.

Der parallele Synchronisationsbus SYNCHRO hat die Aufgabe, Blockadressen, die durch eine Prozessornummer und eine Anforderungsart mit einem Präfix versehen sind, auszusenden, das sind circa 30 bis 40 Bit, je nach den Eigenschaften des Mehrprozessors. Diese Informationen werden außerdem unidirektional übertragen. Ihr Transfer kann auch hier wieder vorteilhafterweise über eine Serienverbindung durchgeführt werden. Der Transfertakt ist weniger kritisch als für die Blöcke, und es können vereinfachte Lösungen ins Auge gefaßt werden, zum Beispiel mittels der von der Firma "A.M.D." hergestellten "TAXI"-Schaltkreise.

Abbildung 19 stellt einen Teil eines synoptischen Schemas einer erfindungsgemäßen Architektur dar, in der mehrere Zentraleinheiten  $UC_k$  ... in Cluster zusammengefaßt sind und sich dieselbe Serienverbindung  $LS_k$  teilen. Zu diesem Zweck hat ein lokaler, dem Cluster zugeordneter Arbitr  $ABL_k$  die Aufgabe, die Zugriffskonflikte mittels Blockadressen-Übertragung zu schlichten, und er teilt sich mit dem Speicherprozessor PGM, der zu diesem Zweck ausgerüstet ist, ein Signal  $busy_k$ , das ständig den freien oder belegten Zustand der Serienverbindung  $LS_k$  angibt. Codierungs- und Decodierungsmittel eines Identifizierungskopfs des betreffenden Prozessors innerhalb eines Clusters sind den Sende- und Empfangslogiken der Datenblöcke zugeordnet.

In dem Fall, wo der gemeinsame Bus BUSA das Übertragungsmittel für die Blockadressen ist, ist die Funktionsweise wie folgt:

Will die Zentraleinheit  $CPU_{k+j}$  einen Blocktransfer vom Hauptspeicher RAM zum Cache-Speicher  $MC_{k+j}$  (Fall dnp, cl, de) oder einen Informations-Schreibvorgang ei vornehmen, dann fordert sie vom lokalen Arbitr  $ABL_k$  den Zugriff auf den gemeinsamen Bus BUSA, der, wenn er an der Reihe ist, dem Arbitr AB die Anforderung weitergibt. Die Zugriffserlaubnis zum

gemeinsamen Bus BUSA wird an die Zentraleinheit  $CPU_{k+j}$  zurückgesandt, und der Transfer erfolgt in der bezüglich Abbildung 10 beschriebenen Weise. Jeder vom Hauptspeicher RAM zum Cache-Speicher  $MC_{k+j}$  übertragene Block muß dann

5 identifiziert werden, da die Reihenfolge der Anforderungen nicht eingehalten wird, weil sie in die Warteschlange AFIFO des Speicher-Verwaltungsprozessors PGM gestellt werden können. Will die Zentraleinheit  $CPU_{k+j}$  einen Blocktransfer vom Cache-Speicher  $MC_{k+j}$  zum Hauptspeicher

10 RAM (Fall maj) vornehmen, dann fordert sie vom lokalen Arbitrer  $ABL_k$  zuerst den Zugriff zu der Serienverbindung  $LS_k$ . Der lokale Arbitrer  $ABL_k$  und der Speicher-Verwaltungsprozessor PGM können sich beide die Serienverbindung  $LS_k$  geben: ein Konflikt wird durch

15 Synchronisation der Modifikation des Signals  $busy_k$  mit dem Signal valid vermieden (der Speicher-Verwaltungsprozessor PGM kann einen Transfer nur während einer Speichertransaktion starten oder erneut starten). Die Belegungserlaubnis der Serienverbindung  $LS_k$  bringt die

20 Zentraleinheit  $CPU_{k+j}$  dazu, ihren Informationsblock zum Speicher-Schieberegister  $RDM_k$  zu übertragen, und dann vom lokalen Arbitrer  $ABL_k$  den Zugriff zum gemeinsamen Bus BUSA zu fordern, um dort die Update-Anforderung vorzunehmen, die nach dem bezüglich Abbildung 10 beschriebenen

25 Algorithmus erfolgt. Das Update-Schreiben kann bewirken, daß in der Warteschlange AFIFO des Speicher-Verwaltungsprozessors PGM ein Block freigemacht wird und ein belegtes Schieberegister  $RDM_j$  gefordert wird. In diesem Fall wird der angeforderte Transfer verzögert und

30 mit dem sich in Gang befindlichen Transfer verkettet.

In dem Fall, wo die Übertragungsmittel für die Blockadressen die Serienverbindungen selbst sind, ist die Funktionsweise, was die Preemption der Serienverbindung betrifft, identisch mit dem vorherigen Fall und, was den

35 allgemeinen Funktionsalgorithmus betrifft, identisch mit dem bezüglich Abbildung 17 beschriebenen.

Zum Beispiel benötigt eine Lese-Anforderung eines Blocks aus der Zentraleinheit  $CPU_{k+j}$  zuerst eine

Zugriffserlaubnis zu der Serienverbindung  $LS_k$ ; diese Erlaubnis wird vom lokalen Arbitr  $ABL_k$  in Übereinstimmung mit dem Speicher-Verwaltungsprozessor PGM gegeben. Die Zugriffserlaubnis führt zum Transfer der Adresse des angeforderten Blocks auf der Serienverbindung  $LS_k$ , die sogleich freigemacht wird: sie steht, wenn nötig, für jede andere Transaktion zur Verfügung. Eine Block-Schreib-Anforderung folgt demselben Protokoll für den Zugang zu der Serienverbindung  $LS_k$ .

In den bezüglich der Abbildungen 1 bis 19 beschriebenen Architekturen gab es genausoviele Speicher-Schieberegister  $RDM_j$  wie Zentraleinheiten  $CPU_j$ : eine Serienverbindung  $LS_j$  ist einem Paar  $(RDM_j, CPU_j)$  statisch zugeordnet gewesen.

Es muß offensichtlich wenigstens eine Serienverbindung  $LS_j$  zwischen einer Zentraleinheit und dem Hauptspeicher RAM geben, die Anzahl der Schieberegister  $RDM_j$  kann jedoch niedriger sein. Tatsächlich ist es nicht möglich, wenn  $tacc$  die Zugriffsdauer auf den Hauptspeicher RAM und  $ttfr$  die Transferdauer eines Blocks ist, daß mehr als  $n = ttfr/tacc$  gleichzeitig belegte Schieberegister beibehalten werden. Zum Beispiel erhält man für  $tacc = 100$  ns und  $ttfr = 1\ 200$  ns:  $n = 12$ .

$Tacc$  und  $ttfr$  sind also charakteristische Leistungskriterien des erfindungsgemäßen Mehrprozessorsystems, und die Installierung von  $n$  Speicher-Schieberegistern  $RDM_j$  ist nur mit einer höheren Anzahl von Serienverbindungen  $LS_j$  kompatibel, und zwar unter der Bedingung, daß eine Logik vom Typ des Verbundnetzes RI zwischen Registern und Verbindungen eingeschoben wird, wobei die Zuordnung eines Speicherregisters  $RDM_j$  zu einer Serienverbindung  $LS_j$  dynamisch vom Speicher-Verwaltungsprozessor PGM durchgeführt wird.

Im übrigen besteht der Hauptspeicher RAM im allgemeinen aus  $m$  parallel angeordneten Speicherbänken  $RAM_1, \dots, RAM_p, RAM_m$ , wobei jede Speicherbank  $n$  Schieberegister  $RDM_j$ , die durch ein Verbundnetz  $RI_p$  mit

allen Serienverbindungen  $LS_j$  verbunden sind, aufweist. Es ist somit möglich, unter dem Vorbehalt, daß die Blockadressen gleichmäßig auf die Speicherbänke  $RAM_p$  verteilt werden, eine theoretische Leistung von  $m \times n$  gleichzeitig aktiven Schieberegistern zu erhalten. Für die gleichmäßige Verteilung der Adressen wird durch herkömmliche Adressenverschachtelungs-Mechanismen gesorgt.

In Abbildung 20a ist eine erfindungsgemäße Architektur teilweise dargestellt, die  $m$  Speicherbänke  $RAM_p$  mit  $n$  Schieberegistern  $RDM_j$  pro Speicherbank und  $q$  Zentraleinheiten  $UC_j$  aufweist. Jede Speicherbank ist vom Typ mit Zufallsreihenfolgezugriff und besitzt einen Dateneingang/-ausgang von der Breite eines Informationsblocks  $b_i$ , wobei dieser Eingang/Ausgang (wie zuvor bei dem Speicher  $RAM$ ) durch einen parallelen Bus mit allen elementaren Registern  $RDM_{1p} \dots RDM_{jp}$  verbunden ist.

Das Verbundnetz ist von einer an sich bekannten Struktur ("cross-bar", "delta", "banyan" ...). Man kann feststellen, daß ein mehrstufiges Netz in dem Maße gut geeignet ist, wo die Dauer des Wegeaufbaus unbedeutend gering im Vergleich mit seiner Belegungsdauer ist (die Transferdauer eines Blocks) und daß nur ein Bit pro Verbindung betroffen ist.

Der Speicher-Verwaltungsprozessor  $PGM$  ist geeignet, um für die dynamische Zuteilung eines Ausgangs zu einem Eingang des Netzes zu sorgen, das heißt, ein Speicher-Schieberegister  $RDM_j$ , und eine Serienverbindung  $LS_i$  in Verbindung zu bringen.

Bestehen die Blockadressen-Übertragungsmittel aus dem gemeinsamen Bus  $BUSA$ , ist die Funktionsweise wie folgt:

Bei einer Lese-Anforderung eines Blocks durch die Zentraleinheit  $CPU_j$ , teilt der betreffende Speicher-Verwaltungsprozessor  $PGM_p$  ein Schieberegister  $RDM_i$  zu, steuert entsprechend das Verbundnetz  $RI$  und initialisiert den Transfer.

Bei einer Schreib-Anforderung eines Blocks durch die Zentraleinheit  $CPU_j$  muß zuvor ein Weg aufgebaut

werden. Dazu wird eine erste Abfrage für den Wegaufbau auf dem gemeinsamen Bus BUSA ausgesandt, gefolgt von der eigentlichen Schreib-Anforderung sofort nach dem Blocktransfer vom Cache-Speicher  $MC_j$  zum Schieberegister  $RDM_j$ . Bei der ersten Anforderung hat der Speicher-Verwaltungsprozessor PGM die Aufgabe, einen Weg zuzuteilen und das Verbundnetz RI zu steuern.

In dem Fall, wo die Blockadressen-Übertragungsmittel die Serienverbindungen selbst sind, muß vor einem Transfer ein Weg aufgebaut werden. Dieses Problem ist mit dem herkömmlichen Problem des Teilens einer Gesamtheit von  $n$  Hilfsmitteln durch  $m$  Benutzer identisch und kann durch herkömmliche Arbitrations-Lösungen von Zugriffskonflikten gelöst werden (Übertragungsprotokolle, zusätzliche Signale).

In dem in Abbildung 5 dargestellten Architekturbeispiel, sind die Schieberegister  $RDM_j$  und  $RDP_j$  und deren Freigabelogiken LV1 und LV2 in einer schnellen Technologie ausgeführt worden, wobei das Ganze durch einen Taktgeber mit der Frequenz  $F$  von mindestens 100 MHz synchronisiert worden ist.

Abbildung 20b stellt als Variante der in Abbildung 20a vorgeschlagenen Architektur eine erfindungsgemäße Lösung dar, bei der jede Serienverbindung  $LS_j$ , die den Prozessor  $CPU_j$  mit allen Speicherbänken verbindet, in  $m$  Serienverbindungen  $LS_{jp}$  aufgeteilt ist und die Punkt zu Punkt den Prozessor  $CPU_j$  mit jeder der Speicherbänke  $RAM_p$  verbindet.

Diese Vorgehensweise hat den folgenden doppelten Vorteil:

- da jede Verbindung eine Punkt zu Punkt-Verbindung ist, kann sie, elektrisch oder für eine Glasfaser, besser angepaßt werden,
- sobald der Verarbeitungs-Prozessor in der Lage ist, Blockanforderungen vorwegzunehmen, was zur Zeit bei den leistungsfähigsten Prozessoren der Fall ist, wird eine zusätzliche Parallelebene erreicht.

Die der Serienverbindung  $LS_j$  auf der Seite des Prozessors  $CPU_j$  zugeordnete Schnittstellenlogik (zuvor mit  $TFR_j$  und  $RDP_j$  bezeichnet) wird dann in  $m$  Exemplare  $I_1 \dots I_p \dots I_m$  vervielfacht. Es kann das Vorhandensein einer

5 Verbindung zur Aufrechterhaltung der Kohärenz der Information eigens für jede Speicherbank  $RAM_p$  festgestellt werden. Die Funktionsweise dieser Verbindung ist analog zu der des Busses SYNCHRO der Abbildung 18.

In den Abbildungen 21a und 21b wurde eine

10 andere RAM-Speicherstruktur dargestellt, die  $2^u$  Speicherebenen aufweist, wobei jede Speicherebene eine Breite von  $t/2^u$  Binärinformationen aufweist (aus Gründen der Klarheit wurden in Abbildung 21a die notwendigen Lese-

15 notwendigen Schreib-Mittel dargestellt). Die Schieberegister  $RDM_j$  oder  $RDP_j$  bestehen aus  $2^u$  elementaren Unterschieberegistern  $RDM_{jp}$  mit einer Kapazität von  $t/2^u$  Bit. Das in Abbildung 20 dargestellte Beispiel ist eine

20 Ausführung mit 8 Speicherebenen ( $u = 3$ ). (Aus Gründen der Klarheit der Zeichnung wurde nur ein einziges, aus allen Unterregister  $RDM_{jp}$  gebildetes Schieberegister  $RDM_j$  dargestellt). Jede Speicherebene  $RAM_p$  enthält gegenüber seiner Zugriffsbreite mehrere elementare Schieberegister  $RDM_{jp}$  und kann mit einer Schiebefrequenz von mindestens

25  $F/2^u$  funktionieren.

Die Funktionsweise des Ganzen beim Lesen ist in Abbildung 21a dargestellt. Ein Block wird synchron aus allen  $2^u$  Speicherebenen gelesen und genauso in die elementaren Register mit dem gleichem Stellenwert geladen.

30 Die Serienausgänge dieser Register sind an die Eingänge eines Multiplexers  $MUXR$ , der in einer schnellen Technologie (ASGA) ausgeführt ist, angeschlossen. Ein genau angepaßter Schaltkreis dieses Typs ist bei "GIGABIT LOGIC" unter der Referenz "10G040" verfügbar und kann ein

35 logisches Signal mit einer Frequenz von 2,7 GHz ausgeben. Er liefert im übrigen einen Taktgeber mit einer durch 8 geteilten Frequenz, der den Schiebetaktgeber der elementaren Register  $RDM_{jp}$  bildet.

Beim Schreiben wird eine symmetrische Funktionsweise, die in Abbildung 21b dargestellt ist, erreicht, und zwar mit einem Multiplex-Schaltkreis DMUXR desselben Herstellers (Referenz "10G41") und mit denselben  
 5 Eigenschaften bezüglich der Leistungsfähigkeit.

Man erreicht so eine Transferfrequenz von 500 MHz mit 8 elementaren Registern, die mit einer Frequenz von  $500/8 = 62,5$  MHz arbeiten, was es ermöglicht, diese elementaren Register in einer konventionelleren  
 10 Technologie zu bauen ("MOS" zum Beispiel).

Die oben angegebenen Multiplexer- und Demultiplexer-Schaltkreise sind in Sätzen von 16, 32, ... Bit kombinierbar. So ist es möglich, bei einer Zuordnung von 16 beziehungsweise 32 62,5-MHz-Speicherebenen,  
 15 Datenraten von 1 und 2 GHz zu erhalten, das bedeutet ein 2 bis 4 mal höheres Leistungsniveau.

Es muß angemerkt werden, daß die Logik TFR auf einem der elementaren Register durchgeführt werden kann und daß die Freigabelogik LV in die "ASGA"-Schaltkreise  
 20 (Ausgang mit offenem Kollektor) integriert ist.

Abbildung 22 stellt die allgemeine Struktur eines Bauteils vom Typ einer integrierten "VLSI"-Schaltkreis dar, die "Serienmultiportspeicher" genannt wird und geeignet ist, ein erfindungsgemäßes  
 25 Mehrprozessorsystem auszurüsten. Dieses Bauteil kann in der zuvor beschriebenen Mehrprozessor-Architektur genutzt werden, und zwar entweder um den Hauptspeicher RAM und die zugeordneten Schieberegister  $RDM_j$  auszuführen oder um jeden Cache-Speicher  $MC_j$  und dessen Schieberegister  $RDP_j$   
 30 auszuführen. Um die Bezeichnungen zu vereinfachen, wurden in der nachfolgenden Beschreibung die Symbole für den Hauptspeicher RAM und die zugeordneten Schieberegister beibehalten.

Die Liste der Stifte dieses Schaltkreises mit den entsprechenden Signalen ist folgende:

-  $adbloc_0$ - $adbloc_{m-1}$ : m Adreßbits eines Blocks  
 bi,

- $\text{admot}_0\text{-admot}_{k-1}$ :  $k$  Wortadressbits in dem Block,
- $\text{numreg}_0\text{-numreg}_{n-1}$ :  $n$  Adressbits eines Registers  $\text{rd}$ ,
- 5     -  $\overline{\text{cs}}$ : "chip select": Auswahlsignal des Schaltkreises,
- $\overline{\text{wr}}$ : "write": Schreibsignal,
- $\overline{\text{rd}}$ : "read": Lesesignal,
- $\text{bit}/\overline{\text{bloc}}$ : Steuersignal der
- 10   Multiportfunktion,
- $\text{normal}/\overline{\text{config}}$ : Signal für Funktionsmodus,
- $\text{data}_0\text{-data}_{l-1}$ :  $l$  Datenbits,
- $h_1\text{-}h_n$ :  $n$  Taktgebersignale,
- $d_1\text{-}d_n$ :  $n$  Datensignale.
- 15     Die Werte  $m$ ,  $n$  und  $l$  sind vom aktuellen Stand der Technologie abhängig. Aktuelle Werte könnten folgende sein:
- $m = 16$  das sind  $2^{16}$  Blöcke  $b_i$  von je 64 Bit (das sind 4 Mbit),
- 20     -  $n = 3$  das sind 8 Register  $\text{rd}$ ,
- $l = 8$  das ist eine Byte-Typ-Parallelschnittstelle,
- $k = 3$  aufgrund des Vorhandenseins von 8 Byte pro Block.
- 25     Das angestrebte Bauteil weist circa fünfzig Stifte auf.
- Diese Schaltung "Multiportserienspeicher" besteht aus einem Schreib-Lese-Speicher mit Zufallsreihenfolgezugriff RAM mit einer vorbestimmten
- 30   Breite  $t$ , die für das Schreiben auf unabhängigen Fronten von einer Breite von  $t/4$  (Beispielwert in Abbildung 22) und  $t/l$  gesteuert werden kann. Die Datenleitungen dieses Speichers RAM sind mit den Eingängen einer Art "Trommel"-Logik BS ("Barrel shifter") oder Multiplexierlogik MT, je
- 35   nach der Version des Bauteils, verbunden, wobei die Multiplexierlogik MT so betrachtet werden kann, daß sie eine Unterbaugruppe der Möglichkeiten der "Trommel"-Logik bietet und somit einfacher in der Ausführung ist. Die

Adressen- und Steuersignale dieses Speichers RAM, das heißt  $cs_i$ ,  $wr_i$ ,  $rd_i$ ,  $adbloc_i$  werden von einer Steuerlogik COM ausgegeben. Diese Logik COM empfängt außerdem die Informationssignale der Stifte  $\overline{cs}$ ,  $\overline{wr}$ ,  $\overline{rd}$ ,  $bit/bloc$ ,  
 5 normal/config und numreg und ist einerseits durch Steuerleitungen "Format" mit der Art "Trommel"-Logik BS und andererseits mit dem Ausgang eines Konfigurationsregisters  $RC_1$  und mit dem Eingang einer Auswahllogik LSR verbunden, die die Signale  $srd_0$ , ...  
 10  $srd_{n-1}$  und  $src_1$ ,  $src_2$ , und  $src_3$  liefert. Die Ausgänge der "Trommel"-Logik BS bilden einen internen Bus zur parallelen Übertragung BUSI, der mit mehreren Schieberegistern  $RD_0$ , ...  $RD_{n-1}$  verbunden ist, und zwar einerseits durch ihre parallelen Eingänge und andererseits  
 15 durch ihre parallelen Ausgänge über die Freigabepuffer  $BV100_0$ , ...  $BV100_{n-1}$  und mit dem parallelen Eingang der Konfigurationsregister  $RC_1$ ,  $RC_2$  ...  $RC_i$ .

Die niederwertigeren 1-Bits des Busses BUSI werden ebenfalls auf den 1-Stiften  $data_0$  ...  $data_{1-1}$   
 20 empfangen. Jedes Schieberegister  $RD_i$  und zugeordnete Verknüpfungen bilden eine funktionelle Einheit  $ELRD_i$ , die durch mehrere logische Elemente gesteuert wird, die eine Setzlogik  $LF_i$  bilden. Jede funktionelle Einheit  $ELRD_i$  weist Verknüpfungen  $ET100_i$  und  $ET101_i$  auf, die auf einem  
 25 ihrer Eingänge mit dem Ausgang  $srd_i$  der Auswahllogik LSR verbunden sind und die auf ihrem anderen Eingang das Signal  $rd_i$  beziehungsweise  $wr_i$  empfangen. Der Ausgang der Verknüpfung  $ET100_i$  ist mit dem Eingang  $load100_i$  des Schieberegisters  $RD_i$  sowie mit dem Eingang  $load101_i$  und  
 30 mit dem Eingang S eines Zählers  $CPT100_i$  beziehungsweise einer Kippschaltung  $B100_i$ , die zu der Setzlogik  $LF_i$  gehört, verbunden. Der Ausgang der Verknüpfung  $ET101_i$  ist mit dem Steuereingang der Freigabepuffer  $BV100_i$  verbunden. Der Ausgang  $di$  ist an den Ausgang einer Verknüpfung  $PL_i$   
 35 angeschlossen, die auf ihrem Dateneingang den Serienausgang des Schieberegisters  $RD_i$  und auf ihrem Steuereingang den Ausgang einer Verknüpfung  $OU100_i$  empfängt. Das aus dem Stift  $h_i$  kommende Signal wird an den

Eingang  $\text{clk100}_i$  des Registers  $\text{RD}_i$  sowie an den Eingang  $\text{down100}_i$  des Zählers  $\text{CPT100}_i$  ausgegeben. Der Ausgang  $\text{zéro100}_i$  des Zählers  $\text{CPT100}_i$  ist mit dem Eingang R der Kippschaltung  $\text{B100}_i$  verbunden.

5 Die Setzlogik  $\text{LF}_i$  weist des weiteren einen Multiplexer  $\text{MUX100}_i$  auf, der auf seinen Dateneingängen die Werte  $t$  und  $t/4$  empfängt. Der Datenausgang des Multiplexers  $\text{MUX100}_i$  ist mit dem Dateneingang des Zählers  $\text{CPT100}_i$  verbunden, und der Auswahlsteuereingang  $\text{sel100}_i$   
 10 des Multiplexers  $\text{MUX100}_i$  ist mit dem Ausgang 1 des Registers  $\text{RC}_i$  verbunden. Der Ausgang Q der Kippschaltung  $\text{B100}_i$  ist mit einem der Eingänge einer Verknüpfung  $\text{ET102}_i$  verbunden, die auf ihrem anderen Eingang das Signal aus dem Stift i eines Registers  $\text{RC}_2$  empfängt. Der Ausgang der  
 15 Verknüpfung  $\text{ET102}_i$  ist mit einem der Eingänge der Verknüpfung  $\text{OU100}_i$  verbunden, die auf ihrem anderen Eingang das Signal aus dem Stift i eines Registers  $\text{RC}_3$  empfängt. Die Ladeeingänge der Register  $\text{RC}_1$ ,  $\text{RC}_2$  und  $\text{RC}_3$  empfangen das Signal  $\text{src}_1$ ,  $\text{src}_2$  beziehungsweise  $\text{src}_3$  aus  
 20 der Auswahllogik LSR .

Dieses Bauteil besitzt eine doppelte Funktionsweise: befindet sich das Signal  $\text{bit}/\text{bloc}$  in dem Zustand "bit", dann funktioniert dieses Bauteil wie ein konventioneller Halbleiterspeicher: die Signale  $\text{adbloc}$ ,  
 25 die den Signalen  $\text{admot}$  zugeordnet sind, bilden den Adreßbus in Worteinheiten (im Beispiel 8 Bit), die Signale  $\overline{\text{cs}}$ ,  $\overline{\text{rd}}$  und  $\overline{\text{wr}}$  haben die normalerweise diesen Signalen zugeordnete Richtung und die Stifte data übertragen die Daten.

30 Intern beim Lesen wird der mit  $\text{adbloc}$  bezeichnete Informationsblock im RAM-Speicher gelesen und dem Eingang der Trommel-Logik BS oder der Multiplexierlogik MT vorgelegt. Die Kombination der Signale  $\text{admot}$  und  $\text{bit}/\text{bloc}$  ermöglichen der Steuerlogik  
 35 COM, der Trommel-Logik BS oder der Multiplexierlogik MT die Signale "format" zu liefern. Das betreffende Wort wird dann rechtsbündig am Ausgang der Trommel-Logik oder

Multiplexierlogik ausgerichtet und so auf den Datenstiften data vorgelegt.

Intern beim Schreiben, wird das auf den Datenleitungen data vorgelegte Wort durch die Trommellogik  
 5 BS oder Multiplexierlogik MT bündig ausgerichtet, und zwar durch dieselben Formatierungssteuersignale wie beim Lesen, gegenüber seiner Position in dem Block. Die Steuerlogik COM sendet dann ein partielles Schreibsignal wri auf den einzigen betreffenden "Teilabschnitt" des Speichers und an  
 10 die von den Signalen adbloc bezeichnete Adresse.

Befindet sich das Signal bit/bloc im Zustand "bloc", dann hängt die Funktionsweise vom Zustand des Signals normal/config ab. Der Modus config programmiert die Konfigurationsregister  $RC_1$ ,  $RC_2$  und  $RC_3$ , die von den  
 15 Signalen numreg adressiert und durch die Datenleitungen data programmiert werden. Das Register  $RC_1$  ermöglicht die Modifikation der Blockgröße:  $t$  und  $t/4$  im Beispiel, das sind 64 Bit und 16 Bit. Intern ist der Funktionsmodus ähnlich dem, der in dem Funktionsmodus "bit" beschrieben  
 20 ist:  $t$  oder  $t/4$  Bits werden auf dem internen Bus BUSI (zum Lesen) bündig oder gegenüber dem betreffenden "Teilabschnitt" des Blocks (für das Schreiben) ausgerichtet. Es können Vielfach-Blockgrößen ins Auge gefaßt werden ( $t$ ,  $t/2$ ,  $t/4$  ...).

Das Register  $RC_3$  ermöglicht für jedes Register die Wahl einer ständigen Funktionsrichtung: entweder für die Eingabe ( $RC3_i = 0$ ) oder für die Ausgabe ( $RC3_i = 1$ ). Durch diese beständige Richtung kann das Bauteil an die Serienverbindungen mit unidirektionalen ständigen Drähten  
 30 angepaßt werden. Das Register  $RC_2$  ermöglicht, unter dem Vorbehalt, daß sich das entsprechende Bit des  $RC_3$  im Logikzustand 0 befindet, für jedes Register die Auswahl eines Funktionsmodus mit wechselnden bidirektionalen Drähten: bei einem RAM-Speicherlesen geht das betreffende  
 35 Schieberegister  $RD_i$  für die Dauer der Übertragung des Blocks in den Ausgabemodus und geht dann im Eingabemodus in den Ruhezustand zurück. Intern wird die Kippschaltung  $B100_i$ , die die Verknüpfung  $PL_i$  steuert, bei einem

- Ladesignal des Registers  $RDM_i$  auf Eins gesetzt und nach dem Transfer der  $t$  oder  $t/4$  Bits auf Null zurückgesetzt, und zwar mittels des Zählers  $CPT100_i$ , der, je nach dem Zustand des Registers  $RC_i$ , auf  $t$  oder  $t/4$  initialisiert
- 5 worden ist und der auf seinem Zähl Eingang die Impulse des Taktgebers  $hi$  empfängt. Bei dem normalen Betriebsmodus für das Lesen (Signal normal/config im Normalzustand) wird der von den Stiften adbloc adressierte Block in das Register  $RD_i$  geladen, das von den Stiften numreg adressiert wird.
- 10 Ist der Block ein Teilblock ( $t/4$ ), dann wird er in der niederwertigeren Position durch die Trommel-Logik BS oder Multiplexierlogik MT auf dem internen Bus BUSI übertragen. Dieser Block wird dann sofort nach Aktivierung des Taktgebersignals  $hi$  übertragen.
- 15 Bei dem normalen Betriebsmodus für das Schreiben wird der Inhalt des von den Stiften numreg adressierten Registers  $RD_i$  in den Speicherblock RAM an der Adresse adbloc geschrieben. Ist der Block ein Teilblock, wird er in der höherwertigeren Position auf dem internen
- 20 Bus BUSI übertragen und dann gegenüber von dem betreffenden "Teilabschnitt" des Blocks durch die Trommel-Logik BS oder die Multiplexierlogik MT bündig ausgerichtet, und schließlich wird ein partielles Schreibsignal wri auf den betreffenden Teilabschnitt gesandt.
- 25 Es muß angemerkt werden, daß, wenn sich ein Teilblock in Betrieb befindet, die Adresse dieses Teilblocks dann in den Block von den Adreßleitungen admot geliefert wird.
- Dieses Bauteil ist perfekt an die
- 30 verschiedenen beschriebenen Architekturvarianten angepaßt. Parallel angeordnet ermöglichen 8, 16 ... Schaltkreise dieses Typs die Ausführung der in den Abbildungen 20a und 20b beschriebenen Vorrichtung. Ist der RAM-Speicher in einer schnellen Technologie ausgeführt, dann kann dieses
- 35 Bauteil auch in dem Cache-Speicher verwendet werden, indem die internen Register eines selben Bauteils gemäß der in den Abbildungen 20a und 20b beschriebenen Vorrichtung multiplexiert werden.

## PATENTANSPRÜCHE

1/ - Mehrprozessorsystem, das einen Hauptspeicher (RAM) aufweist, der in Informationsblöcken (bi) angeordnet ist, Verarbeitungs-Prozessoren (CPU<sub>1</sub> ... CPU<sub>j</sub> ... CPU<sub>n</sub>), einen Cache-Speicher (MC<sub>j</sub>), der mit jedem Verarbeitungs-Prozessor (CPU<sub>j</sub>) verbunden ist und in Informationsblöcken (bi) angeordnet ist, die genauso groß sind, wie die des Hauptspeichers, ein Verzeichnis (RG<sub>j</sub>) und dessen Verwaltungsprozessor (PG<sub>j</sub>), das jedem Cache-Speicher (MC<sub>j</sub>) zugeordnet ist, wobei das besagte Mehrprozessorsystem dadurch gekennzeichnet ist, daß es ausgestattet ist:

. mit mehreren Schieberegistern, sogenannten Speicherregistern (RDM<sub>1</sub> ... RDM<sub>j</sub> ... RDM<sub>n</sub>), wobei jedes dieser Register (RDM<sub>j</sub>) einer Transferlogik (TFR<sub>j</sub>) zugeordnet und so an den Hauptspeicher (RAM) angeschlossen ist, daß innerhalb eines Zyklus dieses Speichers ein Paralleltransfer zum Lesen oder Schreiben eines Informationsblocks (bi) zwischen dem besagten Register und dem besagten Hauptspeicher möglich ist,

. mit Schieberegistern, sogenannten Prozessorregistern (RDP<sub>1</sub> ... RDP<sub>j</sub> ... RDP<sub>n</sub>), wobei jedes Prozessor-Schieberegister (RDP<sub>j</sub>) einer Transferlogik (TFR'<sub>j</sub>) zugeordnet und so mit dem Cache-Speicher (MC<sub>j</sub>) eines Prozessors (CPU<sub>j</sub>) verbunden ist, daß ein Paralleltransfer zum Lesen oder Schreiben eines Informationsblocks (bi) zwischen dem besagten Schieberegister (RDP<sub>j</sub>) und dem besagten Cache-Speicher (MC<sub>j</sub>) möglich ist,

. mit mehreren Serienverbindungen (LS<sub>1</sub> ... LS<sub>j</sub> ... LS<sub>n</sub>), von denen jede ein Speicher-Schieberegister (RDM<sub>j</sub>) mit einem Prozessor-Schieberegister (RDP<sub>j</sub>) verbindet und geeignet ist, den Transfer von Informationsblöcken (bi) zwischen den beiden betroffenen Registern (RDM<sub>j</sub>, RDP<sub>j</sub>) mit einer Transferfrequenz F von mindestens 100 Megahertz zu ermöglichen,

. mit Mitteln für die Übertragung von Blockadressen, die ein Zusatz-Schieberegister (RDC<sub>j</sub>)

- aufweisen, das so auf jeder Serienverbindung ( $LS_j$ ) parallel an das entsprechende Speicher-Schieberegister ( $RDM_j$ ) angeschlossen ist, daß die Übertragung der Adressen über die Serienverbindungen und das Laden dieser Adressen in die besagten Zusatz-Schieberegister ( $RDC_j$ ) möglich ist, wobei ein Zugriffsverwaltungsarbitr (ABM) mit den besagten Zusatz-Schieberegistern ( $RDC_j$ ) und dem Hauptspeicher (RAM) verbunden ist, um die in den besagten Registern ( $RDC_j$ ) enthaltenen Adressen zu holen und die Zugriffs Konflikte zum Hauptspeicher (RAM) zu verwalten, und die jedem Prozessorregister ( $RDP_j$ ) zugeordnete Transferlogik ( $TRF_j$ ) so von dem Verwaltungsprozessor ( $PG_j$ ) gesteuert wird, daß die Übertragung der Adressen über das besagte Prozessorregister ( $RDP_j$ ) möglich ist.
- 2/ - Mehrprozessorsystem, das einen Hauptspeicher (RAM) aufweist, der in Informationsblöcken ( $bi$ ) angeordnet ist, Verarbeitungs-Prozessoren ( $CPU_1 \dots CPU_j \dots CPU_n$ ), einen Cache-Speicher ( $MC_j$ ), der mit jedem Verarbeitungs-Prozessor ( $CPU_j$ ) verbunden ist und in Informationsblöcken ( $bi$ ) angeordnet ist, die genauso groß sind, wie die des Hauptspeichers, ein Verzeichnis ( $RG_j$ ) und dessen Verwaltungsprozessor ( $PG_j$ ), das jedem Cache-Speicher ( $MC_j$ ) zugeordnet ist, wobei das besagte Mehrprozessorsystem dadurch gekennzeichnet ist, daß es
- ausgestattet ist:
- . mit mehreren Schieberegistern, sogenannten Speicherregister ( $RDM_1 \dots RDM_j \dots RDM_n$ ), wobei jedes dieser Register ( $RDM_j$ ) einer Transferlogik ( $TFR_j$ ) zugeordnet und so an den Hauptspeicher (RAM) angeschlossen ist, daß innerhalb eines Zyklus dieses Speichers ein Paralleltransfer zum Lesen oder Schreiben eines Informationsblocks ( $bi$ ) zwischen dem besagten Register und dem besagten Hauptspeicher möglich ist,
  - . mit Schieberegistern, sogenannten Prozessorregistern ( $RDP_1 \dots RDP_j \dots RDP_n$ ), wobei jedes Prozessor-Schieberegister ( $RDP_j$ ) einer Transferlogik ( $TFR'_j$ ) zugeordnet und so mit dem Cache-Speicher ( $MC_j$ ) eines Prozessors ( $CPU_j$ ) verbunden ist, daß ein

Paralleltransfer zum Lesen oder Schreiben eines Informationsblocks ( $bi$ ) zwischen dem besagten Schieberegister ( $RDP_j$ ) und dem besagten Cache-Speicher ( $MC_j$ ) möglich ist,

5                   . mit mehreren Serienverbindungen ( $LS_1 \dots LS_j \dots LS_n$ ), wobei jede davon ein Speicher-Schieberegister ( $RDM_j$ ) mit einem Prozessor-Schieberegister ( $RDP_j$ ) verbindet und geeignet ist, den Transfer von Informationsblöcken ( $bi$ ) zwischen den beiden betroffenen  
10 Registern ( $RDM_j, RDP_j$ ) mit einer Transferfrequenz  $F$  von mindestens 100 Megahertz zu ermöglichen,

                  . mit Mitteln für die Übertragung von Blockadressen, die einen gemeinsamen Bus für die parallele Übertragung von Blockadressen ( $BUSA$ ) aufweisen, der die  
15 Prozessoren ( $CPU_j$ ) mit dem Hauptspeicher ( $RAM$ ) verbindet, und einen Busarbiter ( $AB$ ), der geeignet ist, die Zugriffskonflikte zu dem besagten Bus zu verwalten, wobei der Verwaltungsprozessor ( $PG_j$ ) mit dem gemeinsamen Bus ( $BUSA$ ) so verbunden ist, daß die Übertragung der Adressen  
20 auf dem besagten gemeinsamen Bus möglich ist.

3/ - Mehrprozessorsystem nach einem der Ansprüche 1 oder 2, dadurch gekennzeichnet, daß:

- jedes Speicher-Schieberegister ( $RDM_j$ ) und jedes Prozessor-Schieberegister ( $RDP_j$ ) in zwei Register  
25 aufgeteilt ist, wobei das eine für den Transfer in die eine Richtung und das andere für den Transfer in die andere Richtung spezialisiert ist,

- jede Serienverbindung ( $LS_j$ ) zwei unidirektionale Seriendrähte für einen bit-für-bit-  
30 Transfer aufweist, die das aufgeteilte Speicher-Schieberegister ( $RDM_j$ ) und das entsprechende aufgeteilte Prozessor-Schieberegister ( $RDP_j$ ) miteinander verbinden, wobei diese Drähte an den besagten Registern angeschlossen sind, und der eine den Transfer in die eine Richtung und  
35 der andere den Transfer in die andere Richtung ermöglicht.

4/ - Mehrprozessorsystem nach einem der Ansprüche 1 oder 2, dadurch gekennzeichnet, daß jede Serienverbindung ( $LS_j$ ) einen bidirektionalen Draht für den

bit-für-bit-Transfer aufweist, der an das Speicher-Schieberegister ( $RDM_j$ ) und an das entsprechende Prozessor-Schieberegister ( $RDP_j$ ) angeschlossen ist sowie eine Logik (LV) für die Freigabe der Transferrichtung, sodaß ein  
 5 wechselnder Transfer in beide Richtungen möglich ist.

5/ - Mehrprozessorsystem nach einem der Ansprüche 1, 2, 3 oder 4, das Verwaltungsmittel für zwischen Prozessoren geteilte Daten zur Erhaltung ihrer Kohärenz aufweist.

10 6/ - Mehrprozessorsystem nach Anspruch 5, dadurch gekennzeichnet, daß die Verwaltungsmittel der geteilten Daten folgendes aufweisen:

. einen speziellen Bus für die parallele Wortübertragung (BUSD), der die Prozessoren ( $CPU_j$ ) mit dem  
 15 Hauptspeicher (RAM) verbindet,

. eine Partitionslogik ( $LP_j$ ), die jedem Prozessor ( $CPU_j$ ) zugeordnet ist und die geeignet ist, die Adressen der geteilten und der nicht geteilten Daten zu unterscheiden, um sie auf den Adressen-Übertragungsmitteln  
 20 mit ihrer Identifizierung zu übertragen,

. eine Decodierungslogik (DEC), die dem Hauptspeicher (RAM) zugeordnet ist und die geeignet ist, die Adressen mit ihrer Identifizierung aufzunehmen und die Daten am Speicherausgang weiterzuleiten, entweder, was die  
 25 nicht geteilten Daten betrifft, zum entsprechenden Speicher-Schieberegister ( $RDM_j$ ) oder, was die geteilten Daten betrifft, zu dem speziellen Bus für die Wortübertragung (BUSD).

7/ - Mehrprozessorsystem nach Anspruch 5, dadurch gekennzeichnet, daß die Verwaltungsmittel der  
 30 geteilten Daten einerseits einen speziellen Bus für die parallele Wortübertragung (BUSD) und einen speziellen gemeinsamen Bus für die Wortadressenübertragung (BUSAM) aufweisen, die die Prozessoren ( $CPU_j$ ) und den  
 35 Hauptspeicher (RAM) miteinander verbinden und andererseits eine jedem Prozessor ( $CPU_j$ ) zugeordnete Partitionslogik ( $LP_j$ ) aufweisen, die geeignet ist, die Adressen der geteilten Daten und die der nicht geteilten Daten zu

unterscheiden und die ersten zu dem speziellen gemeinsamen Bus (BUSAM) und die zweiten zu den Mitteln für die Übertragung von Blockadressen weiterzuleiten.

8/ - Mehrprozessorsystem nach den Ansprüchen 1  
 5 und 5 zusammen, dadurch gekennzeichnet, daß die Verwaltungsmittel der geteilten Daten einen dem Hauptspeicher (RAM) zugeordneten Speicherverwaltungs-Prozessor (PGM) aufweisen sowie einen jedem Verarbeitungs-Prozessor (CPU<sub>j</sub>) und dem entsprechenden  
 10 Verwaltungsverzeichnis (RG<sub>j</sub>) zugeordneten Prozessor für die Aufrechterhaltung der Kohärenz der geteilten Daten (PMC<sub>j</sub>), wobei jeder Prozessor zur Aufrechterhaltung der Kohärenz (PMC<sub>j</sub>) an einen von dem Speicherverwaltungs-Prozessor (PGM) gesteuerten Synchronisationsbus (SYNCHRO)  
 15 so angeschlossen ist, daß bei Entdecken einer Blockadresse ein Update des Hauptspeichers (RAM) und des zugeordneten Cache-Speichers (MC<sub>j</sub>) möglich ist und bei jeder Adressenentnahme aus den Zusatz-Schieberegistern (RDC<sub>j</sub>) ein Update des Hauptspeichers (RAM) und der Cache-Speicher  
 20 (MC<sub>j</sub>).

9/ - Mehrprozessorsystem nach den Ansprüchen 2 und 5 zusammen, dadurch gekennzeichnet, daß die Verwaltungsmittel der geteilten Daten einen dem Hauptspeicher (RAM) zugeordneten Speicherverwaltungs-Prozessor (PGM) aufweisen sowie einen jedem Verarbeitungs-Prozessor (CPU<sub>j</sub>) und dem entsprechenden  
 25 Verwaltungsverzeichnis (RG<sub>j</sub>) zugeordneten Buspion-Prozessor (PE<sub>j</sub>), wobei jeder Buspion-Prozessor (PE<sub>j</sub>) und der Speicherverwaltungs-Prozessor (PGM) an den Adreßbus  
 30 (BUSA) angeschlossen ist, um die auf dem besagten Bus übertragenen Blockadressen zu überwachen beziehungsweise zu verarbeiten, damit bei Entdecken einer in dem zugeordneten Verzeichnis (RG<sub>j</sub>) vorhandenen Blockadresse ein Update des Hauptspeichers (RAM) und des zugeordneten  
 35 Cache-Speichers (MC<sub>j</sub>) möglich ist.

10/ - Mehrprozessorsystem nach den Ansprüchen 2 und 5 zusammen, dadurch gekennzeichnet, daß die Verwaltungsmittel der geteilten Daten einen dem

Hauptspeicher (RAM) zugeordneten Speicherverwaltungs-Prozessor (PGM) und einen Prozessor zur Aufrechterhaltung der Kohärenz der geteilten Daten (PMC<sub>j</sub>) aufweisen, der jedem Verarbeitungs-Prozessor (CPU<sub>j</sub>) und dem  
 5 entsprechenden Verwaltungsverzeichnis (RG<sub>j</sub>) zugeordnet ist, wobei jeder Prozessor zur Aufrechterhaltung der Kohärenz (PMC<sub>j</sub>) an einen Synchronisationsbus (SYNCHRO) angeschlossen ist, der von dem Speicherverwaltungs-Prozessor (PGM) so gesteuert wird, daß bei Entdecken einer  
 10 Blockadresse ein Update des Hauptspeichers (RAM) und des zugeordneten Cache-Speichers (MC<sub>j</sub>) und bei jeder Adressenentnahme auf dem gemeinsamen Adreßbus (BUSA) ein Update des Hauptspeichers (RAM) und der Cache-Speicher (MC<sub>j</sub>) möglich ist.

15 11/ - Mehrprozessorsystem nach einem der Ansprüche 1 bis 10, dadurch gekennzeichnet, daß:

- mehrere, einem bestimmten Satz von Prozessoren (CPU<sub>k</sub>, CPU<sub>k+1</sub> ...) entsprechende Prozessor-Schieberegister (RDP<sub>k</sub>, RDP<sub>k+1</sub> ...) parallel an eine selbe  
 20 Serienverbindung (LS<sub>k</sub>) angeschlossen sind, wobei jedem Satz von Prozessoren (CPU<sub>k</sub>, CPU<sub>k+1</sub> ...) ein lokaler Arbiter (ABL<sub>k</sub>) zugeordnet ist, um die Zugriffskonflikte zu der Serienverbindung (LS<sub>k</sub>) zu lösen,

- ein Speicherverwaltungs-Prozessor (PGM) mit  
 25 den Mitteln für die Übertragung von Blockadressen und mit dem Hauptspeicher (RAM) verbunden ist und Codierungsmittel besitzt, um jedem Informationsblock (bi) einen Kopf zur Identifizierung des betroffenen Prozessors aus jedem Satz der Prozessoren (CPU<sub>k</sub>, CPU<sub>k+1</sub> ...), die sich eine  
 30 Datenserienverbindung (LS<sub>k</sub>) teilen, zuzuordnen,

- die Verwaltungsprozessoren (PG<sub>k</sub>, PG<sub>k+1</sub> ...),  
 die den Cache-Speichern (MC<sub>k</sub>, MC<sub>k+1</sub> ...) der Prozessoren aus dem zuvor genannten Satz (CPU<sub>k</sub>, CPU<sub>k+1</sub> ...) zugeordnet sind, Mittel für die Decodierung des Identifizierungskopfs  
 35 aufweisen.

12/ - Mehrprozessorsystem nach einem der Ansprüche 1 bis 11, dadurch gekennzeichnet, daß jedes Speicher-Schieberegister (RDM<sub>j</sub>) statisch an eine dem

besagten Register spezifisch zugeordnete Serienverbindung ( $LS_j$ ) angeschlossen ist.

13/ - Mehrprozessorsystem nach einem der Ansprüche 1 bis 11, dadurch gekennzeichnet, daß:

5       - ein Speicherverwaltungs-Prozessor (PGM) dem Hauptspeicher (RAM) zugeordnet ist und eine Logik (ALLOC) für die Zuordnung der Speicher-Schieberegister zu den Serienverbindungen aufweist,

10       - die Speicher-Schieberegister ( $RDM_1 \dots RDM_j \dots RDM_n$ ) dynamisch an die Serienverbindungen ( $LS_1 \dots LS_j \dots$ ) angeschlossen sind, und zwar mittels eines Verbundnetzes (RI), das vom Speicherverwaltungs-Prozessor (PGM) gesteuert wird.

15       14/ - Mehrprozessorsystem nach einem der Ansprüche 1 bis 13, in dem der Hauptspeicher (RAM) aus  $m$  parallel angeordneten Speicherbänken ( $RAM_1 \dots RAM_p \dots RAM_m$ ) besteht, dadurch gekennzeichnet, daß jedes Speicher-Schieberegister ( $RDM_j$ ) aus  $m$  elementaren Schieberegistern ( $RDM_{j1} \dots RDM_{jp} \dots RDM_{jm}$ ) besteht, die parallel mit der  
20       entsprechenden Serienverbindung ( $LS_j$ ) verbunden sind, wobei jedes elementare Register ( $RDM_{jp}$ ) an eine Speicherbank ( $RAM_p$ ) so angeschlossen ist, daß innerhalb eines Zyklus der besagten Speicherbank ein paralleler Transfer zum Lesen oder Schreiben eines Informationsblocks  
25       (bi) zwischen dem besagten elementaren Register und der besagten Speicherbank möglich ist.

15/ - Mehrprozessorsystem nach Anspruch 14, in dem jede Serienverbindung ( $LS_j$ ) in  $m$  Serienverbindungen ( $LS_{jp}$ ) aufgeteilt ist, die Punkt zu Punkt jeden Prozessor  
30       (CPU<sub>j</sub>) mit dem elementaren Schieberegister ( $RDM_{jp}$ ) verbinden.

16/ - Mehrprozessorsystem nach Anspruch 14, in dem auf jede Speicherbank ( $RAM_p$ ) mit Zufallsreihenfolge zugegriffen wird und das einen Dateneingang/-ausgang von  
35       einer einem Informationsblock (bi) entsprechenden Breite besitzt, dadurch gekennzeichnet, daß der besagte Eingang/Ausgang, jeder Speicherbank ( $RAM_p$ ) durch einen

parallelen Bus mit den elementaren Registern ( $RDM_{1p} \dots RDM_{jp}$ ) verbunden ist.

17/ - Mehrprozessorsystem nach einem der Ansprüche 14, 15 oder 16, das von einem Taktgeber mit einer Frequenz  $F$  von mindestens 100 Megahertz synchronisiert wird, dadurch gekennzeichnet, daß jedes elementare Speicher-Schieberegister ( $RDM_{jp}$ ) und jedes Prozessor-Schieberegister ( $RDP_j$ ) eine Schiebefrequenz von mindestens  $F$  haben kann.

18/ - Mehrprozessorsystem nach einem der Ansprüche 14, 15 oder 16, das von einem Taktgeber mit einer Frequenz  $F$  von mindestens 100 Megahertz synchronisiert wird, dadurch gekennzeichnet, daß jedes elementare Speicher-Schieberegister und/oder jedes Prozessor-Schieberegister aus mehreren  $2^u$  multiplexierten Unterregistern ( $RDM_{jp}$ ,  $RDP_{jp}$ ) besteht, wobei jedes eine Schiebefrequenz von mindestens  $F/2^u$  haben kann.

19/ - Verfahren zum Informationsaustausch zwischen einem in Informationsblöcken ( $bi$ ) angeordneten Hauptspeicher (RAM) und Prozessoren ( $CPU_1 \dots CPU_j \dots CPU_n$ ), wobei jeder mit einem in Blöcken von derselben Größe ( $bi$ ) angeordneten Cache-Speicher ( $MC_j$ ) und mit einem Verzeichnis ( $RG_j$ ) und dessen Verwaltungsprozessor ( $PG_j$ ) ausgestattet ist, so daß der Austausch zwischen Hauptspeicher (RAM) und jedem Prozessor ( $CPU_j$ ) über den Cache-Speicher ( $MC_j$ ) dieses letzteren erfolgt, wobei das besagte Verfahren dadurch gekennzeichnet ist:

- daß jeder Transfer eines Informationsblocks ( $bi$ ) vom Hauptspeicher (RAM) zum Cache-Speicher ( $MC_j$ ) eines gegebenen Prozessors ( $CPU_j$ ) mit einer Transferfrequenz  $F$  von mindestens 100 Megahertz erfolgt und darin besteht:

innerhalb eines Hauptspeicher-Zyklus den Block ( $bi$ ) von dem besagten Hauptspeicher (RAM) zu einem Speicher-Schieberegister ( $RDM_j$ ) von der Größe eines Blocks zu übertragen, wobei das Speicher-Schieberegister eines von mehreren an den Hauptspeicher angeschlossenen Schieberegistern ( $RDM_1 \dots RDM_j, \dots RDM_n$ ) ist,

. den Inhalt des Speicher-Schieberegisters ( $RDM_j$ ) auf einer Serienverbindung ( $LS_j$ ) zu einem Prozessor-Schieberegister ( $RDP_j$ ) von der gleichen Kapazität zu übertragen, das dem Cache-Speicher ( $MC_j$ ) des betreffenden Prozessors ( $CPU_j$ ) zugeordnet ist,

. den Inhalt des besagten Prozessor-Schieberegisters ( $RDP_j$ ) zu dem besagten Cache-Speicher ( $MC_j$ ) zu übertragen,  
 - daß der Adressentransfer eines Informationsblocks mit der Transferfrequenz  $F$  über die Serienverbindungen erfolgt.

20/ - Verfahren zum Informationsaustausch zwischen einem in Informationsblöcken ( $bi$ ) angeordneten Hauptspeicher (RAM) und Prozessoren ( $CPU_1 \dots CPU_j \dots CPU_n$ ), wobei jeder mit einem Cache-Speicher ( $MC_j$ ), der in Blöcken von der gleichen Kapazität ( $bi$ ) angeordnet ist, und mit einem Verzeichnis ( $RG_j$ ) und dessen Verwaltungsprozessor ( $PG_j$ ) ausgestattet ist, so daß der Austausch zwischen Hauptspeicher (RAM) und jedem Prozessor ( $CPU_j$ ) über den Cache-Speicher ( $MC_j$ ) dieses letzteren erfolgt, wobei das besagte Verfahren dadurch gekennzeichnet ist, daß:

- jeder Transfer eines Informationsblocks ( $bi$ ) vom Hauptspeicher (RAM) zum Cache-Speicher ( $MC_j$ ) eines gegebenen Prozessors ( $CPU_j$ ) mit einer Transferfrequenz  $F$  von mindestens 100 Megahertz erfolgt und darin besteht:

. innerhalb eines Hauptspeicher-Zyklus den Block ( $bi$ ) von dem besagten Hauptspeicher (RAM) zu einem Speicher-Schieberegister ( $RDM_j$ ) von der Größe eines Blocks zu übertragen, wobei das Speicher-Schieberegister eines von mehreren an den Hauptspeicher angeschlossenen Schieberegistern ( $RDM_1 \dots RDM_j \dots RDM_n$ ) ist,

. den Inhalt des Speicher-Schieberegisters ( $RDM_j$ ) auf einer Serienverbindung ( $LS_j$ ) zu einem Prozessor-Schieberegister ( $RDP_j$ ) von der gleichen Kapazität, das dem Cache-Speicher ( $MC_j$ ) des betreffenden Prozessors ( $CPU_j$ ) zugeordnet ist, zu übertragen,

. den Inhalt des besagten Prozessor-Schieberegisters ( $RDP_j$ ) zu dem besagten Cache-Speicher ( $MC_j$ ) zu übertragen,

- 5 - daß der Adressentransfer eines Informationsblocks mittels eines gemeinsamen parallelen Adreßbusses (BUSA) erfolgt.

21/ - Verfahren zum Informationsaustausch zwischen einem in Informationsblöcken ( $bi$ ) angeordneten Hauptspeicher (RAM) und Prozessoren ( $CPU_1 \dots CPU_j \dots$   
10  $CPU_n$ ), wobei jeder mit einem Cache-Speicher ( $MC_j$ ), der in Blöcken der gleichen Kapazität ( $bi$ ) angeordnet ist, und mit einem Verzeichnis ( $RG_j$ ) und dessen Verwaltungsprozessor ( $PG_j$ ) ausgestattet ist, so daß der Austausch zwischen Hauptspeicher (RAM) und jedem Prozessor  
15 ( $CPU_j$ ) über den Cache-Speicher ( $MC_j$ ) dieses letzteren erfolgt, wobei das besagte Verfahren dadurch gekennzeichnet ist:

- daß jeder Transfer eines Informationsblocks ( $bi$ ) vom Cache-Speicher ( $MC_j$ ) eines gegebenen Prozessors ( $CPU_j$ ) zum  
20 Hauptspeicher (RAM) mit einer Transferfrequenz  $F$  von mindestens 100 Megahertz erfolgt und darin besteht :

. den Block ( $bi$ ) des besagten betreffenden Cache-Speichers ( $MC_j$ ) zu einem dem besagten Cache-Speicher ( $MC_j$ ) zugeordneten Prozessor-Schieberegister ( $RDP_j$ ) von  
25 der Größe eines Blocks zu übertragen,

. den Inhalt des Prozessor-Schieberegisters ( $RDP_j$ ) auf einer Serienverbindung ( $LS_j$ ) zu einem Speicher-Schieberegister ( $RDM_j$ ) mit der gleichen Kapazität zu übertragen, das dem betreffenden Prozessor der an den  
30 Hauptspeicher (RAM) angeschlossenen Schieberegistern ( $RDM_1 \dots RDM_j \dots RDM_n$ ) zugeordnet ist,

. innerhalb eines Zyklus des Hauptspeichers den Inhalt des Speicher-Schieberegisters ( $RDM_j$ ) zu dem besagten Hauptspeicher (RAM) zu übertragen,

- 35 - daß der Adressentransfer eines Informationsblocks mit der Transferfrequenz  $F$  über die Serienverbindungen erfolgt.

22/ - Verfahren zum Informationsaustausch zwischen einem in Informationsblöcken (bi) angeordneten Hauptspeicher (RAM) und Prozessoren ( $CPU_1 \dots CPU_j \dots CPU_n$ ), wobei jeder mit einem Cache-Speicher ( $MC_j$ ), der in  
 5 Blöcken der gleichen Kapazität (bi) angeordnet ist, und mit einem Verzeichnis ( $RG_j$ ) und seinem Verwaltungsprozessor ( $PG_j$ ) ausgestattet ist, so daß der Austausch zwischen Hauptspeicher (RAM) und jedem Prozessor ( $CPU_j$ ) über den Cache-Speicher ( $MC_j$ ) dieses letzteren  
 10 erfolgt, wobei das besagte Verfahren dadurch gekennzeichnet ist, daß:

- jeder Transfer eines Informationsblocks (bi) vom Cache-Speicher ( $MC_j$ ) eines gegebenen Prozessors ( $CPU_j$ ) zum Hauptspeicher (RAM) mit einer Transferfrequenz F von  
 15 mindestens 100 Megahertz erfolgt und darin besteht :

. den Block (bi) des besagten betreffenden Cache-Speichers ( $MC_j$ ) zu einem dem besagten Cache-Speicher ( $MC_j$ ) zugeordneten Prozessor-Schieberegister ( $RDP_j$ ) von der Größe eines Blocks zu übertragen,

20 . den Inhalt des Prozessor-Schieberegisters ( $RDP_j$ ) auf einer Serienverbindung ( $LS_j$ ) zu einem Speicher-Schieberegister ( $RDM_j$ ) der gleichen Kapazität zu übertragen, das dem betreffenden Prozessor der an den Hauptspeicher (RAM) angeschlossenen Schieberegistern ( $RDM_1$   
 25 ...  $RDM_j$  ...  $RDM_n$ ) zugeordnet ist,

. innerhalb eines Zyklus des Hauptspeichers den Inhalt des Speicher-Schieberegisters ( $RDM_j$ ) zu dem besagten Hauptspeicher (RAM) zu übertragen,

- der Adressentransfer eines Informationsblocks mittels  
 30 eines gemeinsamen parallelen Adreßbusses (BUSA) erfolgt.

23/ - Bauteil für einen Serienmultiportspeicher, das geeignet ist, ein Mehrprozessorsystem nach einem der Ansprüche 1 bis 18 auszurüsten, dadurch gekennzeichnet, daß es aus einem  
 35 integrierten Schaltkreis besteht, der einen Speicher mit Zufallsreihenfolgezugriff (RAM) von vorbestimmter Breite, die einem Informationsblock (bi) entspricht, aufweist, mehrere Schieberegister ( $RDM_1 \dots RDM_j \dots RDM_n$ ), jedes

mit einer Kapazität, die der Breite des Speichers entspricht, einen parallelen internen Bus (BUSI), der den Zugang des Speichers und die Schieberegister miteinander verbindet, eine Logik zur Auswahl eines Schieberegisters (LSR), die geeignet ist, die Verbindung auf dem internen Bus zwischen dem Speicher und einem vorbestimmten Schieberegister freizugeben und mehrere externe Eingangs-/Ausgangsstifte (adblöc, admot, numreg, cs, wr, rd, bitbloc, normal/config, hi, di) für den Adreßeingang zum Speicher (RAM), für den Adreßeingang zu der Auswahllogik (LSR), für den Eingang und die Freigabe von Lese- oder Schreib-Transferbefehlen eines Informationsblocks (bi) zwischen dem Speicher (RAM) und den Schieberegistern (RDM<sub>j</sub>), für den Eingang eines Taktgebersignals zu jedem Schieberegister (RDM<sub>j</sub>), für den bit-für-bit-Eingang eines Informationsblocks (bi) zu jedem Schieberegister (RDM<sub>j</sub>) und für den bit-für-bit-Ausgang eines Informationsblocks von jedem Schieberegister (RDM<sub>j</sub>).

24/ - Bauteil nach Anspruch 23, dadurch gekennzeichnet, daß es mindestens ein Konfigurationsregister (RC<sub>1</sub>, RC<sub>2</sub> ...) aufweist, das Programmierungseingänge besitzt, wobei jedes Konfigurationsregister mit einer Setzlogik (LF) verbunden ist, die an den Speicher (RAM) und an die Schieberegister (RDM<sub>j</sub>) angeschlossen ist, um die Zustände des besagten Speichers und der besagten Schieberegister zu setzen.

25/ - Bauteil nach Anspruch 24, das die Wahl der Größe der verarbeiteten Informationsblöcke (bi) ermöglicht, dadurch gekennzeichnet, daß:

- der Speicher (RAM) in kombinierbare Bereiche unterteilt ist, um die Speicherung der verschiedenen möglichen Größen von Informationsblöcken zu ermöglichen,
- jedes Schieberegister (RDM<sub>j</sub>) in kombinierbare Teilabschnitte unterteilt ist, um das Laden der verschiedenen möglichen Größen von Informationsblöcken zu ermöglichen, mit Abzweigungen, die für das jeder Größe entsprechende Verschieben sorgen,

- der interne Bus (BUSI) mit einer Multiplexierlogik (MT) ausgestattet ist, um die Transfers von Informationsblöcken (bi) der verschiedenen Größen zwischen den Kombinationen von Speicherbereichen (RAM) und den entsprechenden Kombinationen der Schieberegister-  
 5 Teilabschnitte (RDM<sub>j</sub>) zu ermöglichen,

- ein Konfigurationsregister (RC<sub>1</sub>) mit einer der Anzahl der möglichen Blockgrößen entsprechenden Kapazität vorgesehen ist,

10 - die mit dem Register (RC<sub>1</sub>) verbundene Setzlogik (LF) eine Logikeinheit aufweist, die geeignet ist, die Multiplexierlogik (MT) zu steuern, um die Transfers von Informationsblöcken (bi) in einer gegebenen Größe freizugeben, die dem im Konfigurationsregister (RC<sub>1</sub>)  
 15 enthaltenen Parameter entspricht.

26/ - Bauteil nach einem der Ansprüche 24 oder 25, dadurch gekennzeichnet, daß:

- der Eingang und der Ausgang von jedem Schieberegister (RDM<sub>j</sub>) mittels einer Verknüpfung (PL<sub>j</sub>) mit demselben externen Stift verbunden sind,  
 20

- ein Konfigurationsregister (RC<sub>2</sub>) mit einer der Anzahl der Schieberegister (RDM<sub>j</sub>) entsprechenden Kapazität vorgesehen ist,

- die mit dem Konfigurationsregister (RC<sub>2</sub>) verbundene Setzlogik (LF) eine Logikeinheit aufweist, die geeignet ist, die Verknüpfungen (PL<sub>j</sub>) zu steuern, um die Funktionsweise von jedem Schieberegister (RDM<sub>j</sub>) in den Eingangs- oder Ausgangsmodus zu setzen, und zwar abhängig von einem in dem Konfigurationsregister (RC<sub>2</sub>) enthaltenen  
 25 Bit, das dem besagten Schieberegister (RDM<sub>j</sub>) zugeordnet ist.  
 30

27/ - Bauteil nach einem der Ansprüche 24, 25 oder 26, dadurch gekennzeichnet, daß:

- der Eingang und der Ausgang von jedem Schieberegister (RDM<sub>j</sub>) mittels einer Verknüpfung (PL<sub>j</sub>) mit demselben externen Stift verbunden ist,  
 35

- ein Konfigurationsregister ( $RC_3$ ) mit einer der Anzahl von Schieberegistern ( $RDM_j$ ) entsprechenden Kapazität vorgesehen ist,

- die mit dem Konfigurationsregister ( $RC_3$ ) verbundene Setzlogik (LF) eine Logikeinheit aufweist, die mit der Lese-Steuerung des Speichers (RAM) verbunden ist und die geeignet ist, jede Verknüpfung ( $PL_j$ ) zu steuern, entweder im Ausgangsmodus beim Lesen des Speichers (Transfer des RAM-Speichers zum entsprechenden Register ( $RDM_j$ )) während der ganzen Dauer des vollständigen Entladens des besagten Schieberegisters ( $RDM_j$ ) oder die restliche Zeit im Eingangsmodus.

28/ - Bauteil nach einem der Ansprüche 24, 25, 26 oder 27, dadurch gekennzeichnet, daß es einen externen Eingangsstift (bit/bloc) aufweist, einen oder mehrere externe Eingangs-/Ausgangsstifte für Daten (data), eine Steuerlogik (COM), die mit dem Eingangsstift (bit/bloc), mit den Eingangs-/Ausgangsstiften (data), mit dem Speicher (RAM) und mit der Auswahllogik (LSR) verbunden ist und die geeignet ist, je nach Zustand des Eingangs (bit/bloc), entweder die Transfers von Informationsblöcken (bi) zwischen Speicher (RAM) und Schieberegistern ( $RDM_j$ ) oder die Transfers von Bit direkt zwischen dem Speicher (RAM) und den Stiften (data) zu bewirken.

29/ - Bauteil nach den Ansprüchen 24 und 28 zusammen, dadurch gekennzeichnet, daß die Konfigurationsregister ( $RC_1$ ,  $RC_2$  ...) verbunden sind:

- einerseits mit der Auswahllogik (LSR), die geeignet ist, die besagten Konfigurationsregister für die den besagten Registern zugeordneten vorbestimmten Adressen auszuwählen,

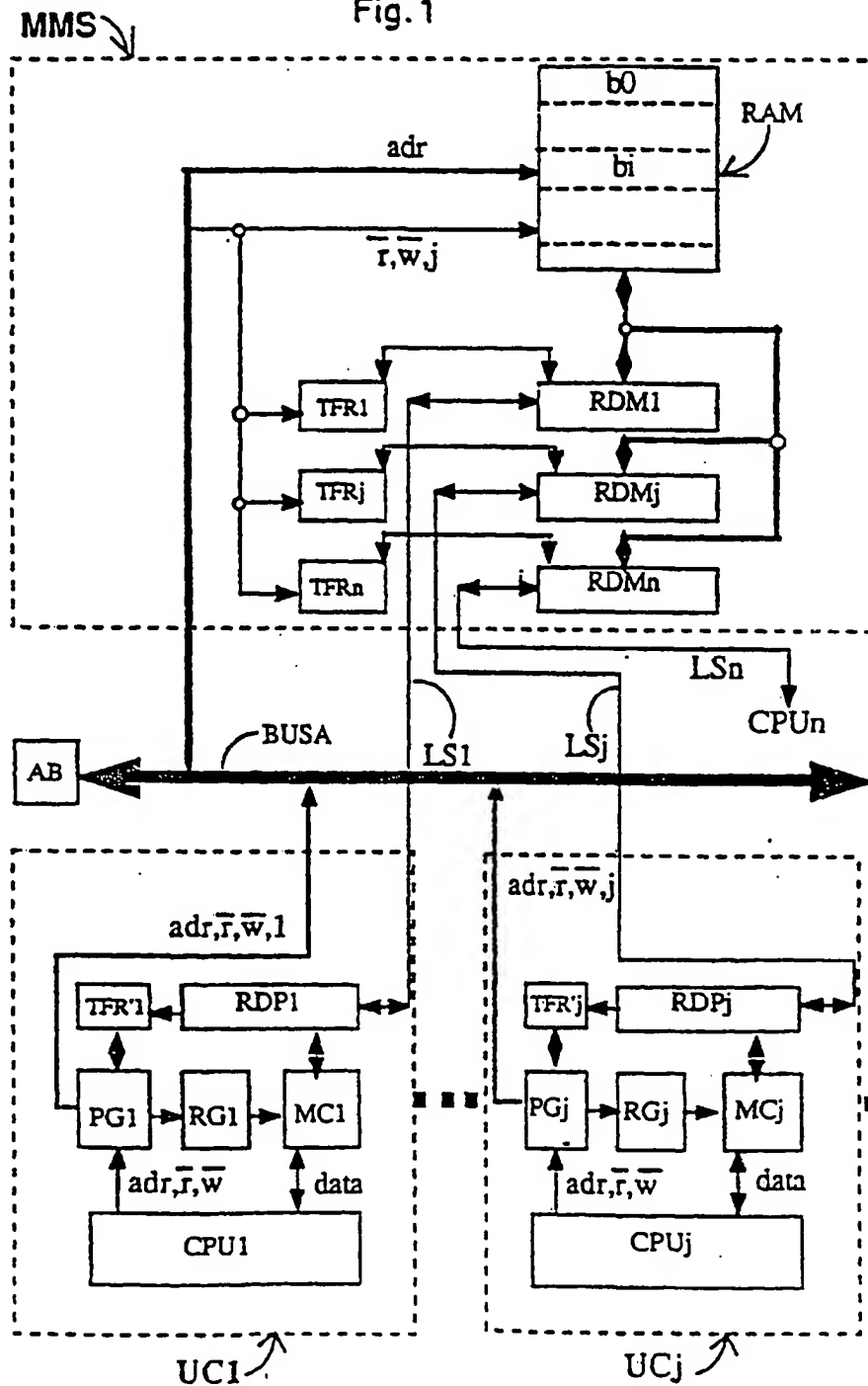
- andererseits, mit der Steuerlogik (COM), die geeignet ist, die von den Eingangs-/Ausgangsstiften (data) kommenden Daten zu den besagten Konfigurationsregistern zu übertragen, um diese letzteren zu programmieren.

30/ - Bauteil nach einem der Ansprüche 23 bis 29, in dem auf dem internen Bus (BUSI), der den Zugang zum Speicher (RAM) mit den Schieberegistern ( $RDM_j$ ) verbindet,

eine Logik vom Typ "Trommel" (BS) ("barrel shifter") zwischengeschaltet ist, die geeignet ist, ein Ringshiften auf den Bit von jedem Informationsblock durchzuführen, wobei die besagte Logik (BS) einen Eingang für die  
5 Steuerung des Verschiebungsschritts - in Worteinheiten - besitzt, der an Eingangsstiften (admot) angeschlossen ist.

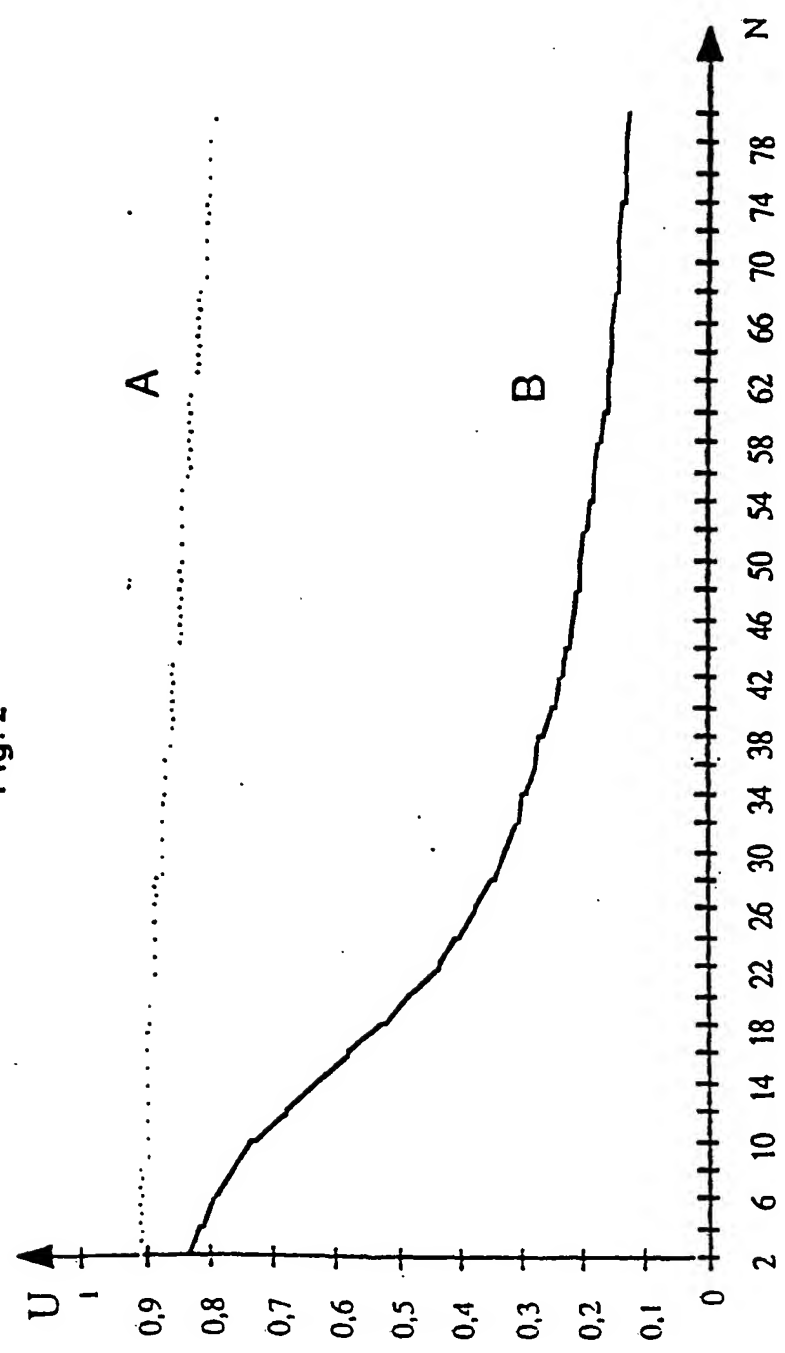
EP 0 346 420

Abb. 1/22  
Fig. 1



2/22

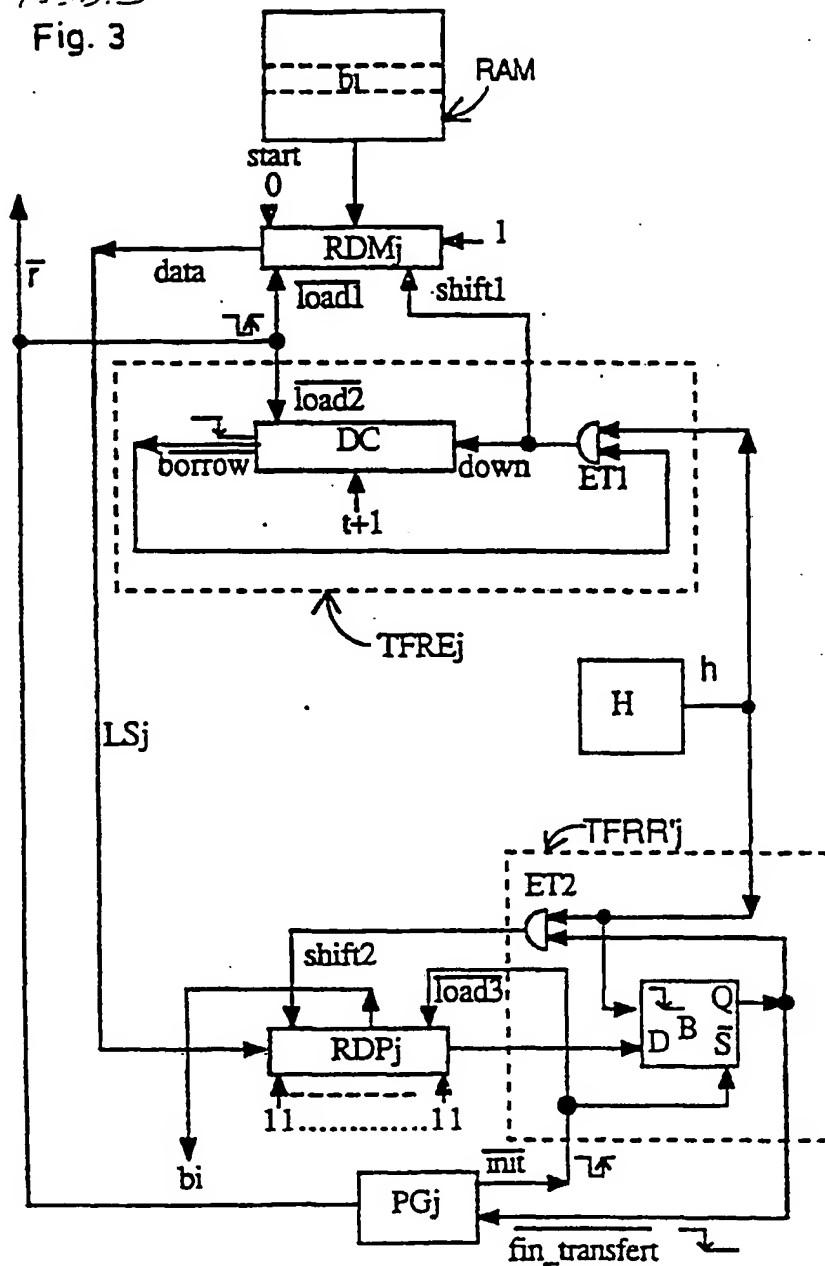
Abbildung  
Fig. 2



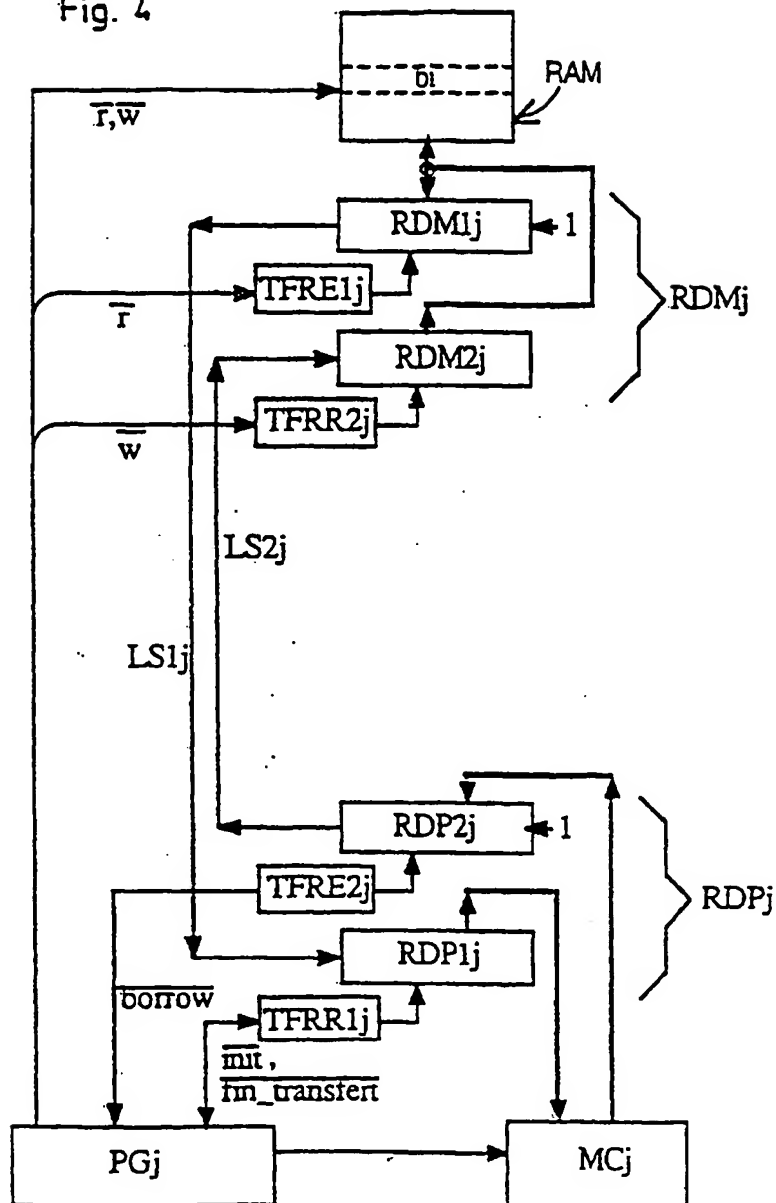
3/22

Ann. 3

Fig. 3

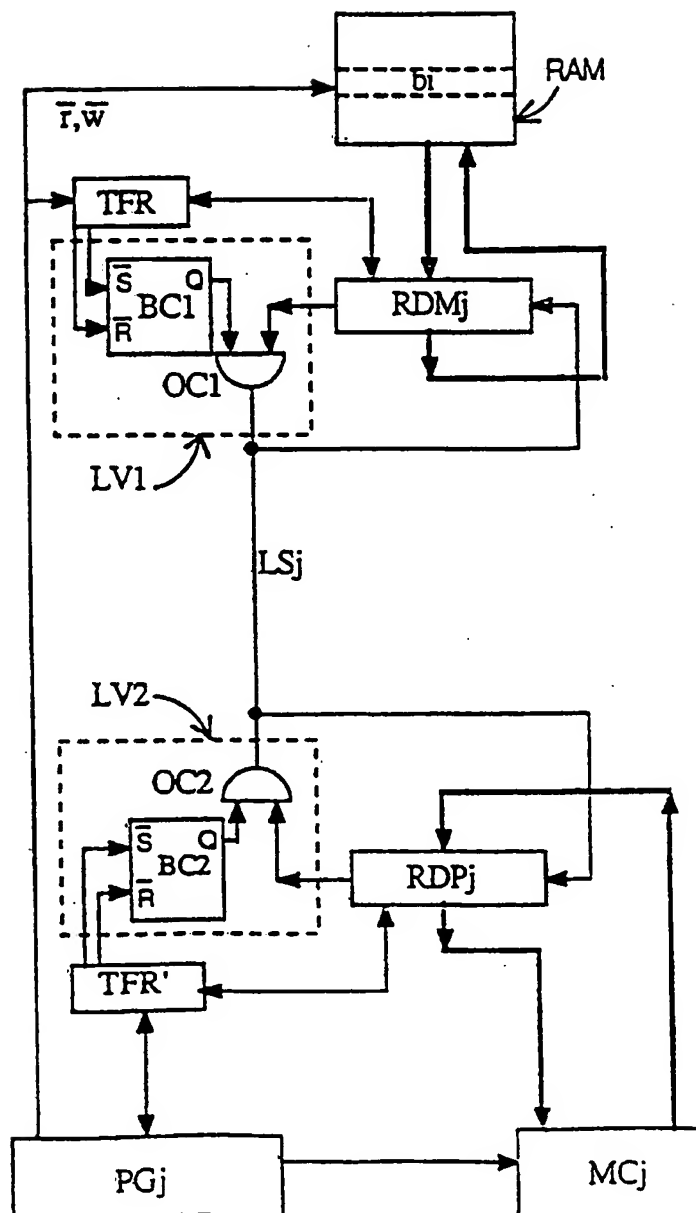


4 / 22

Abb. 4  
Fig. 4

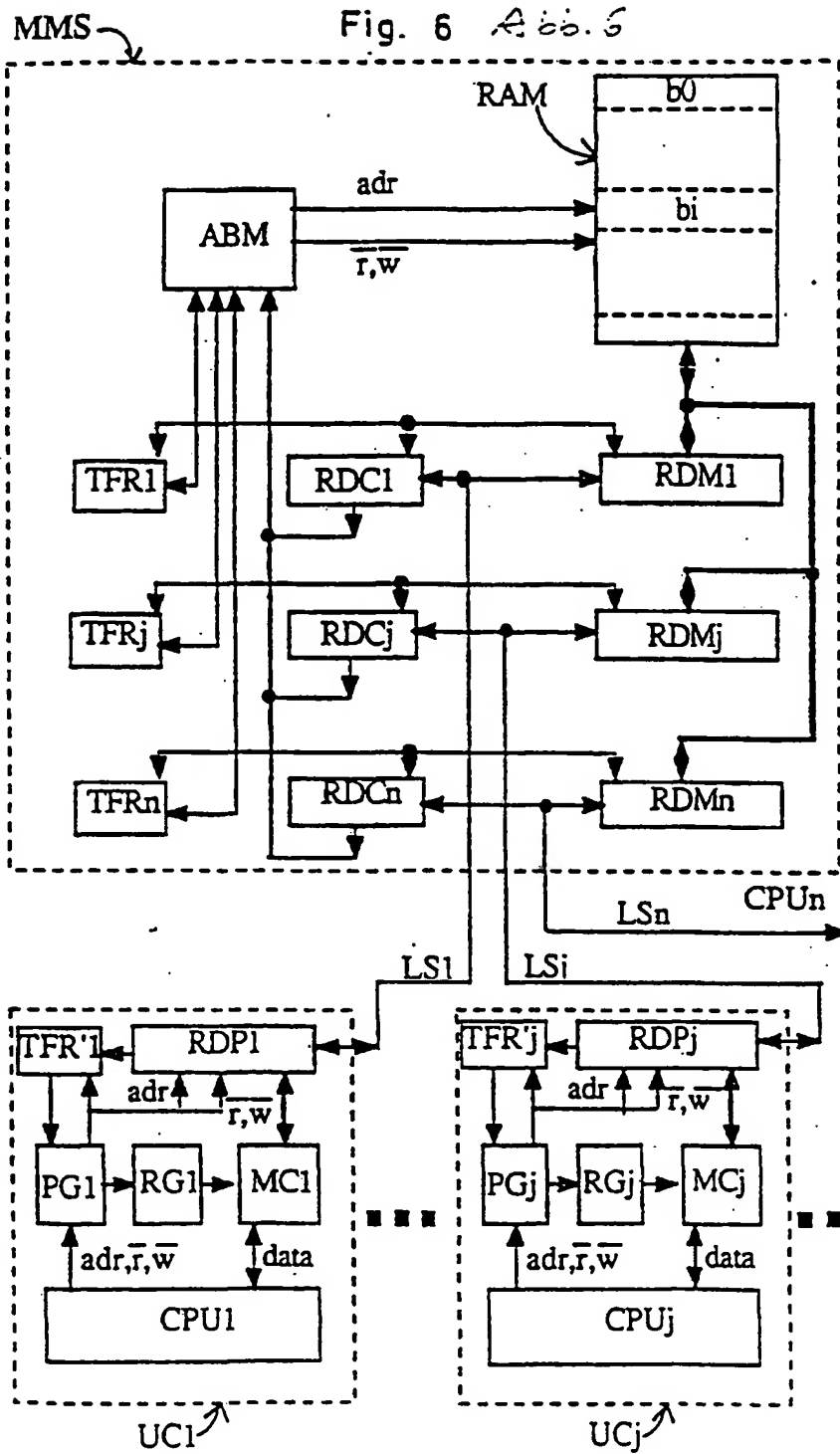
5/22

Abb. 5  
Fig. 5

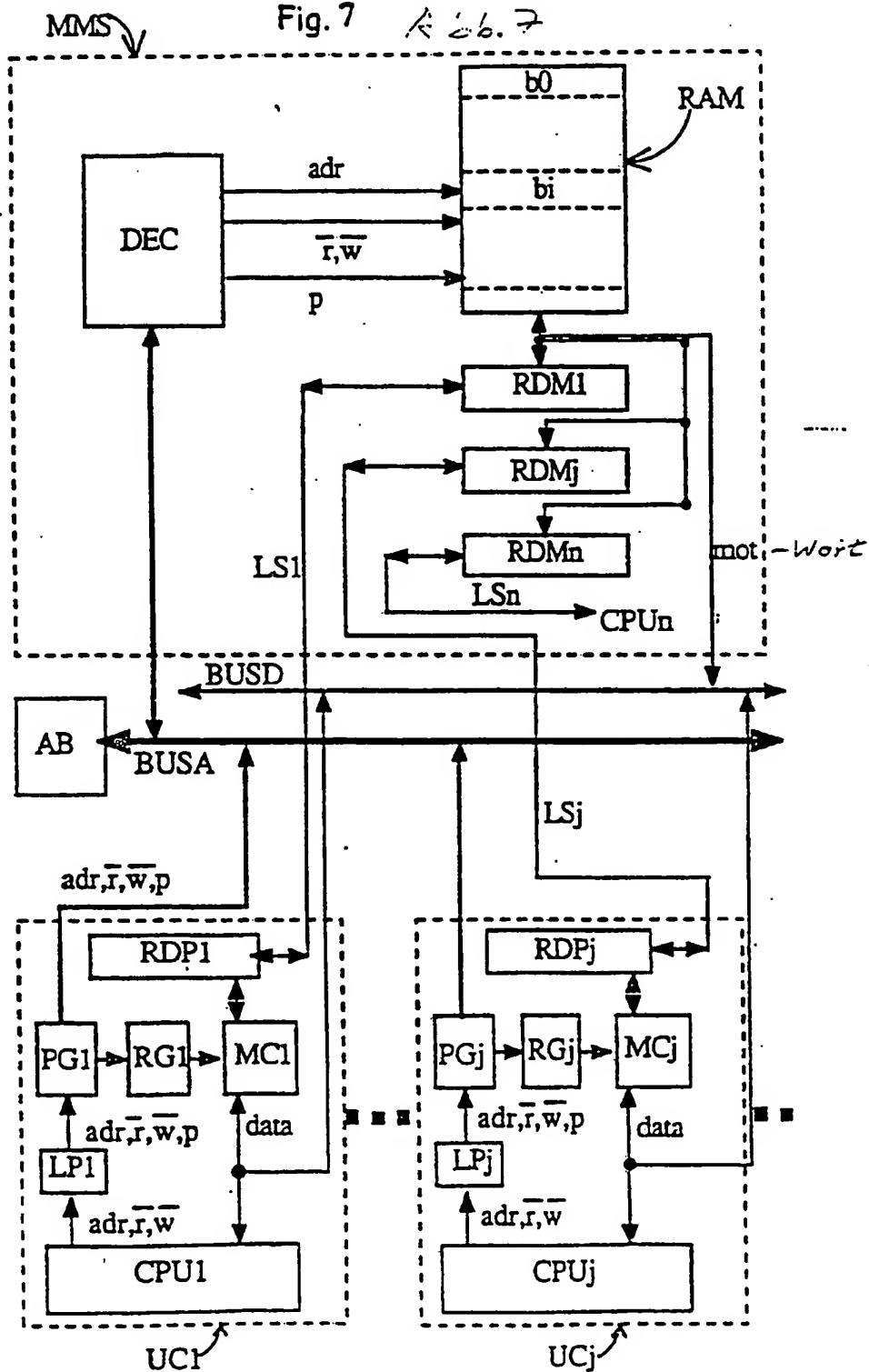


6/22

Fig. 6 Abb. 6

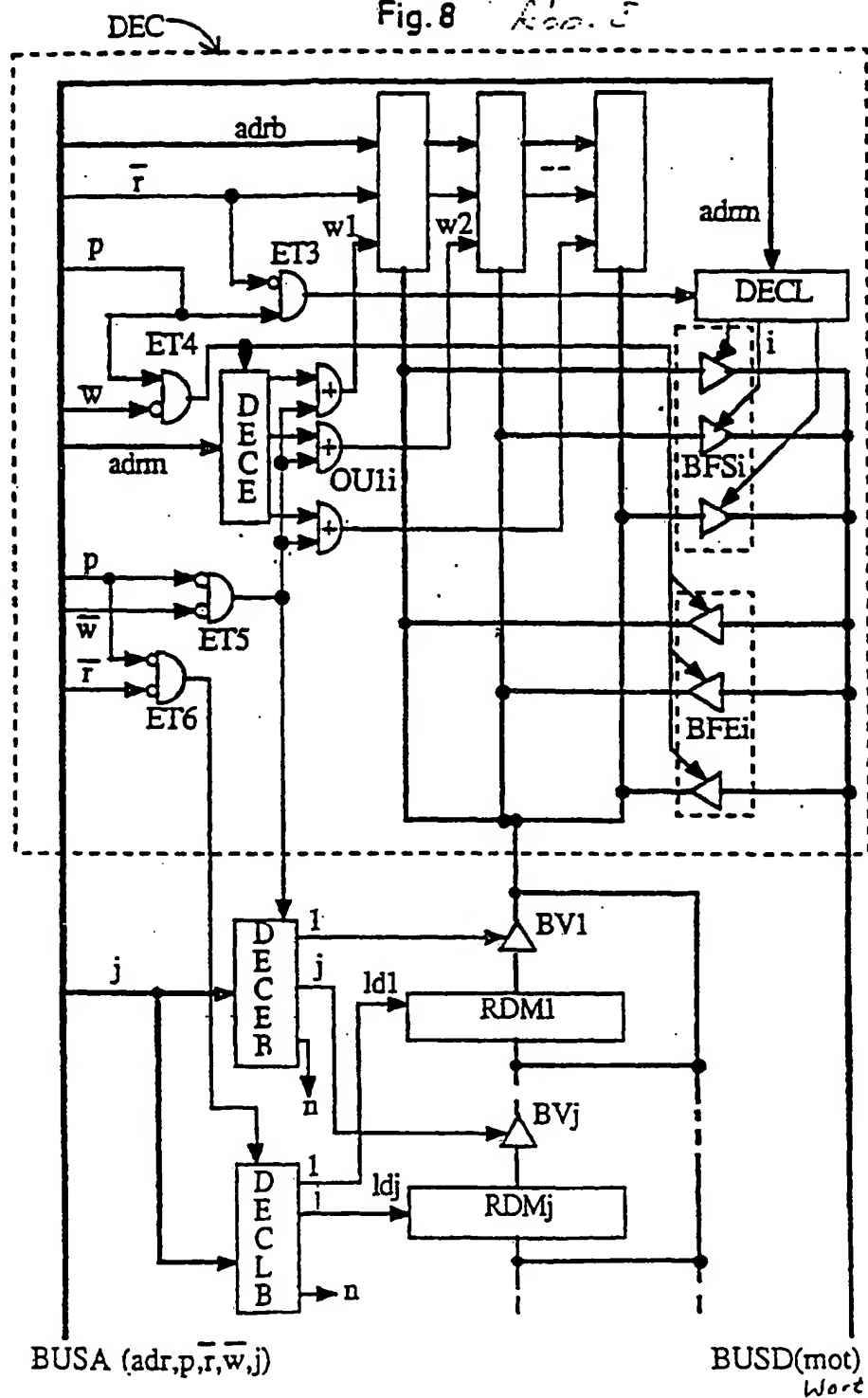


7/22

Fig. 7 *Abb. 7*

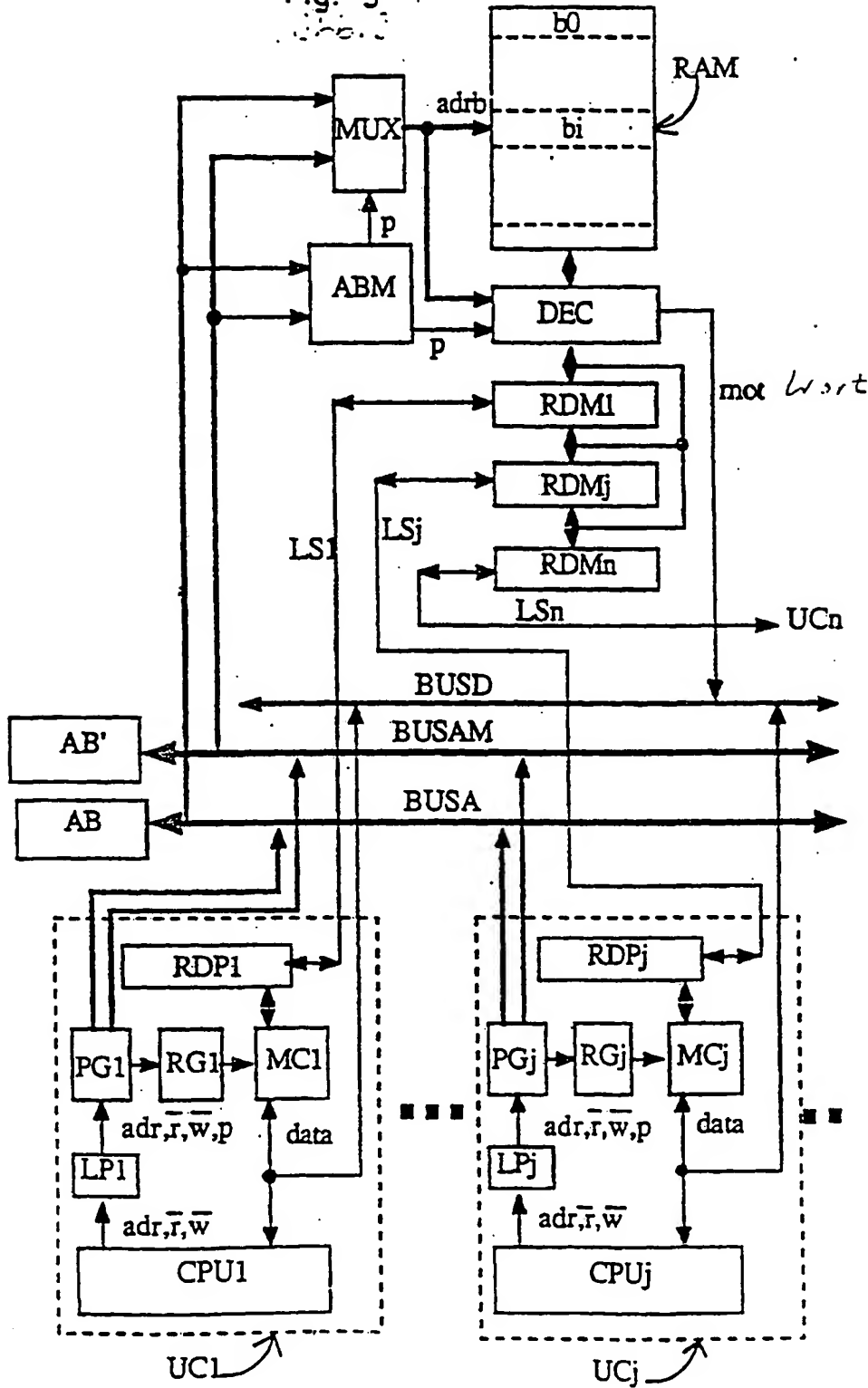
8/22

Fig. 8 *Tab. 5*



9/22

Fig. 9

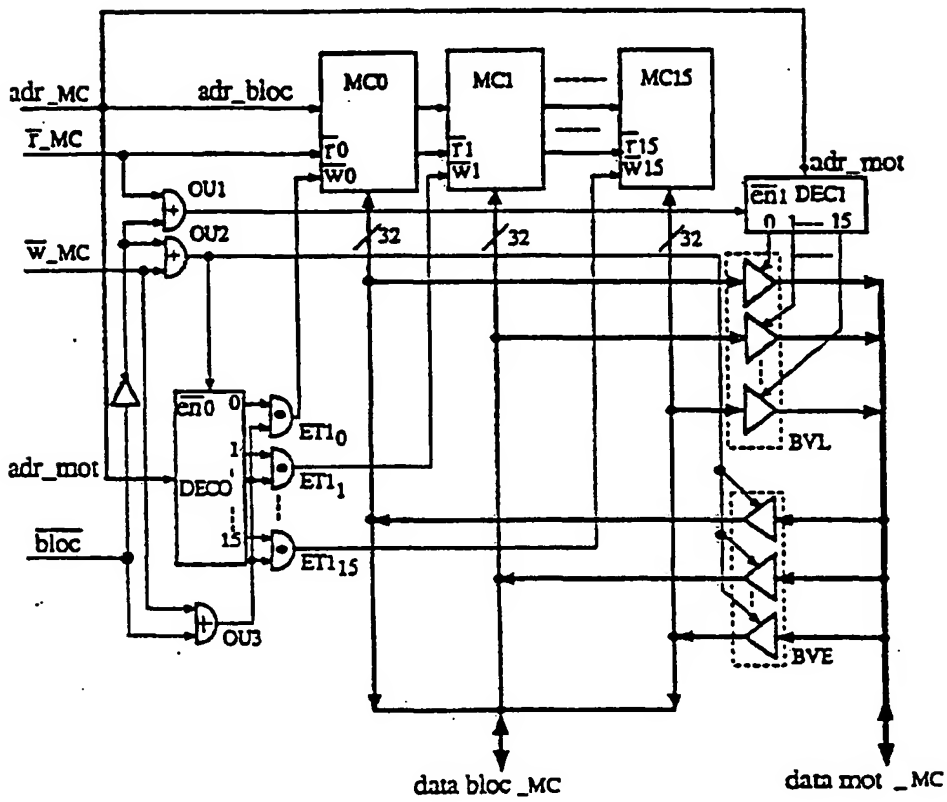




11 / 22

Fig. 11

126. 11



12/22

Fig. 12a

Abb. 12a

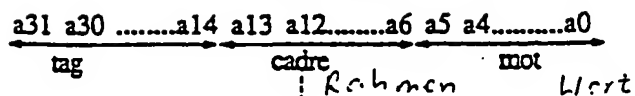


Fig. 12b -

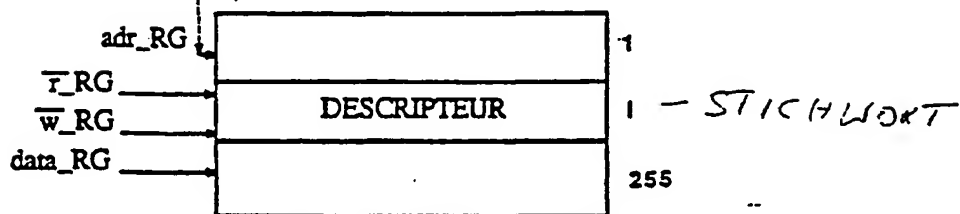


Fig. 12c

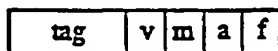
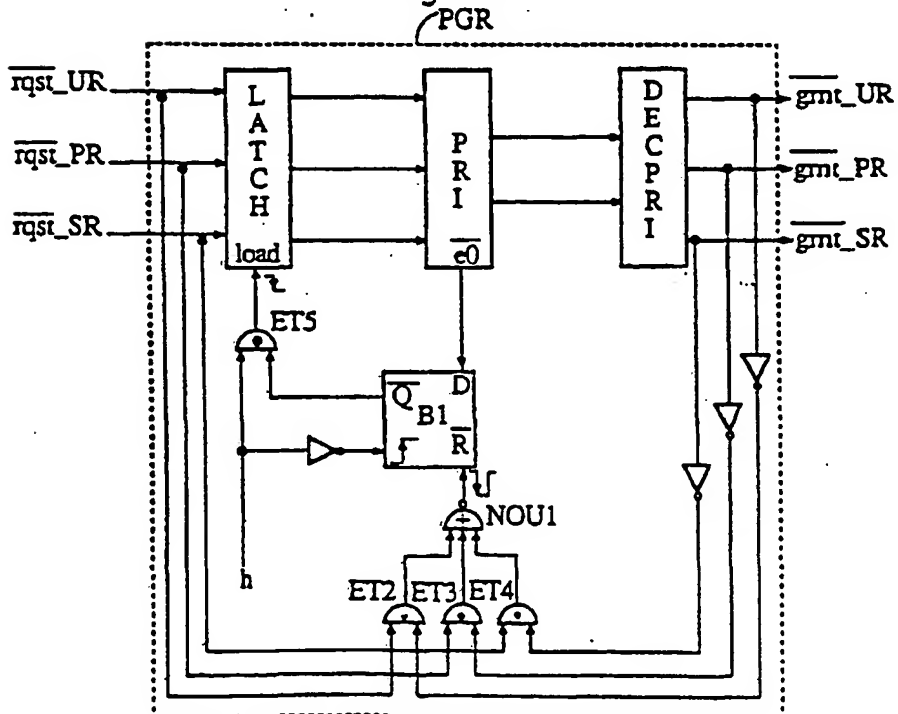
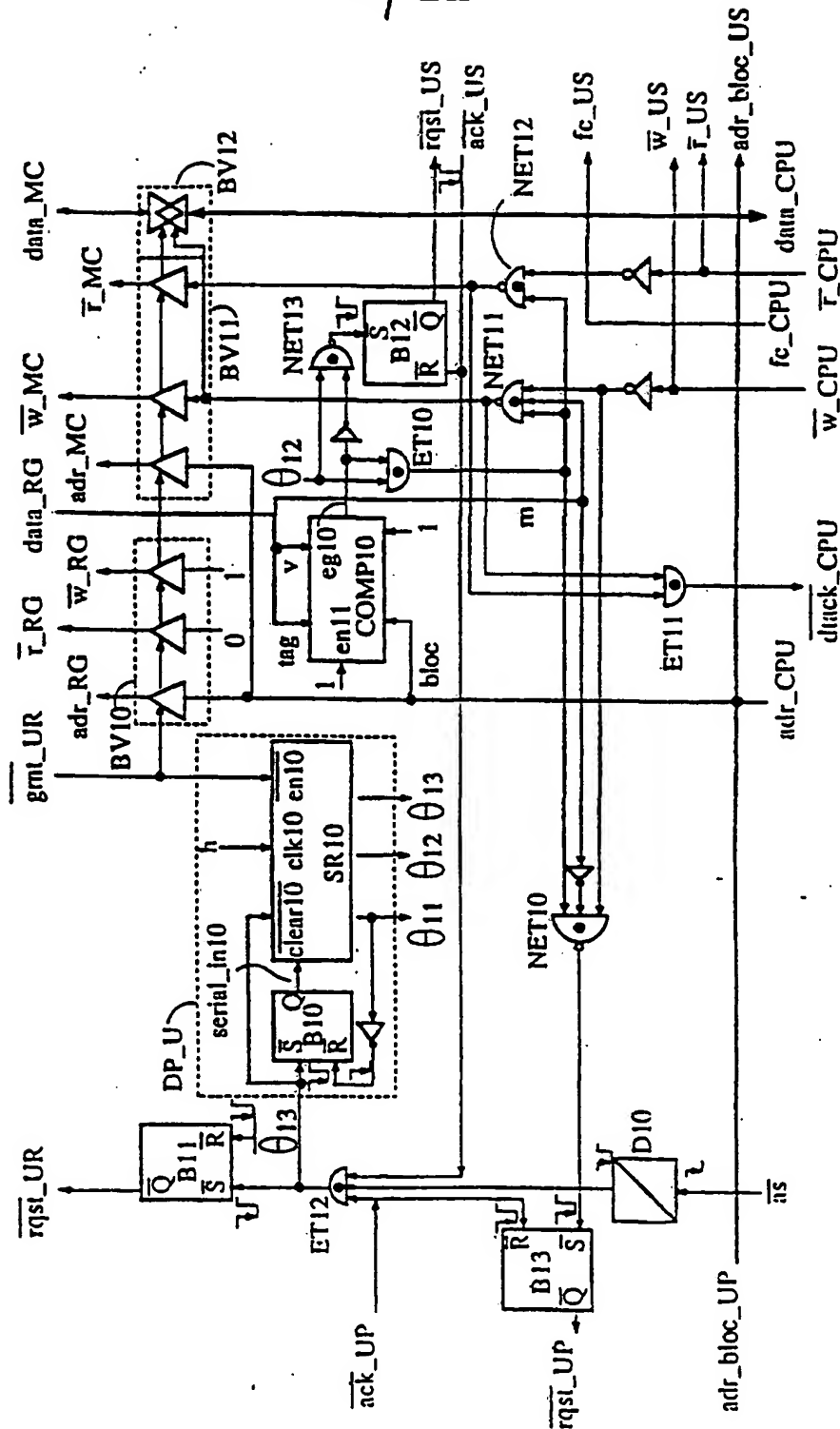


Fig. 12d



13/22

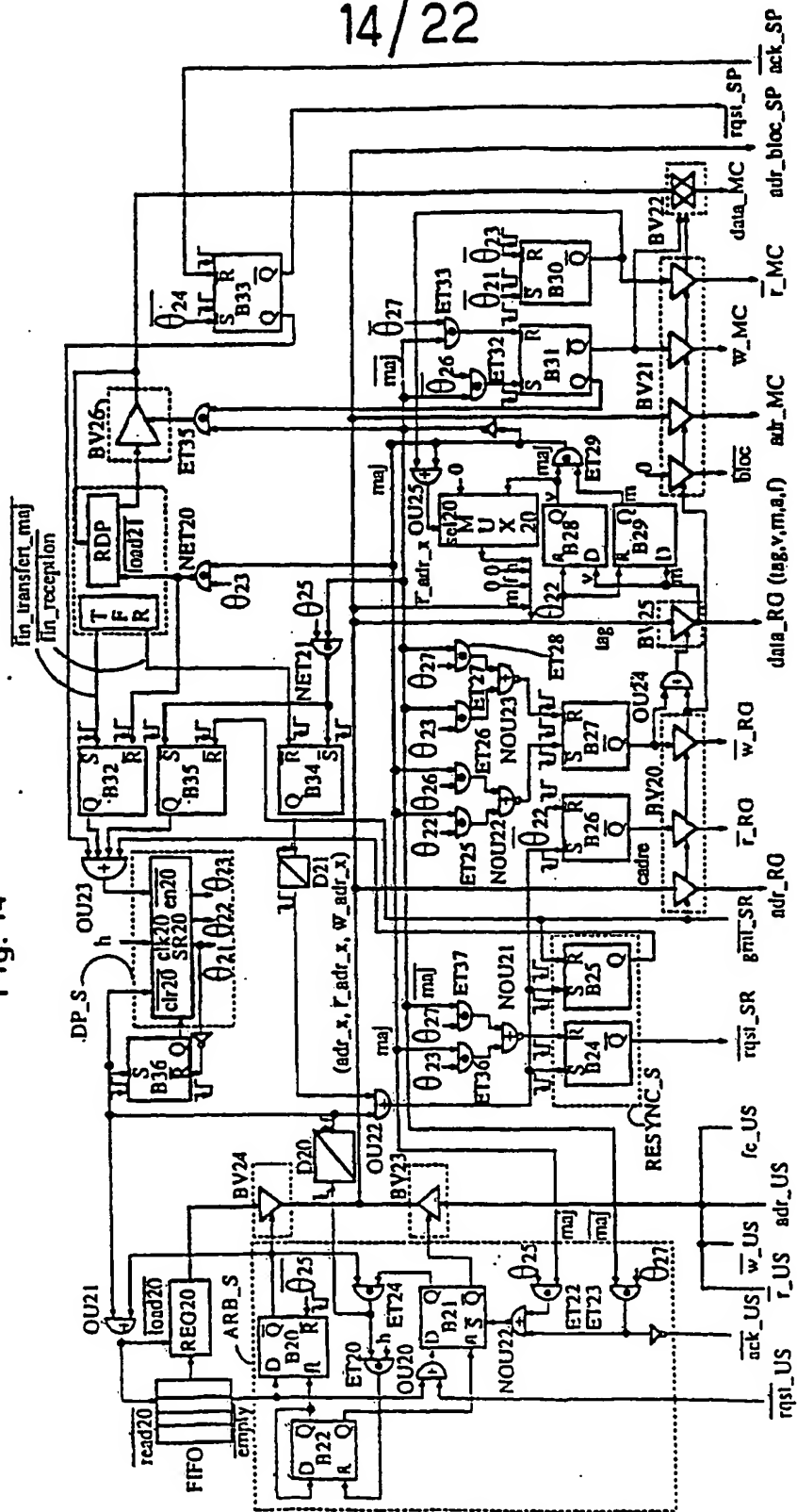
Fig. 13



14/22

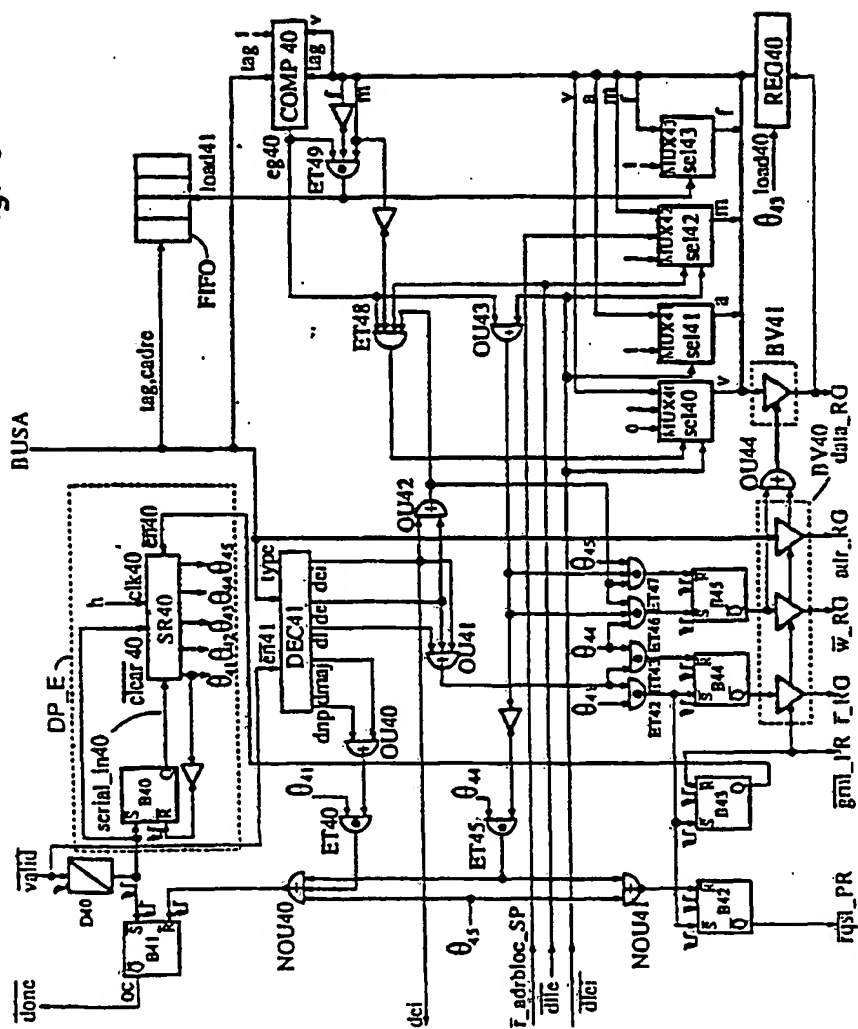
Ann. 14

Fig. 14



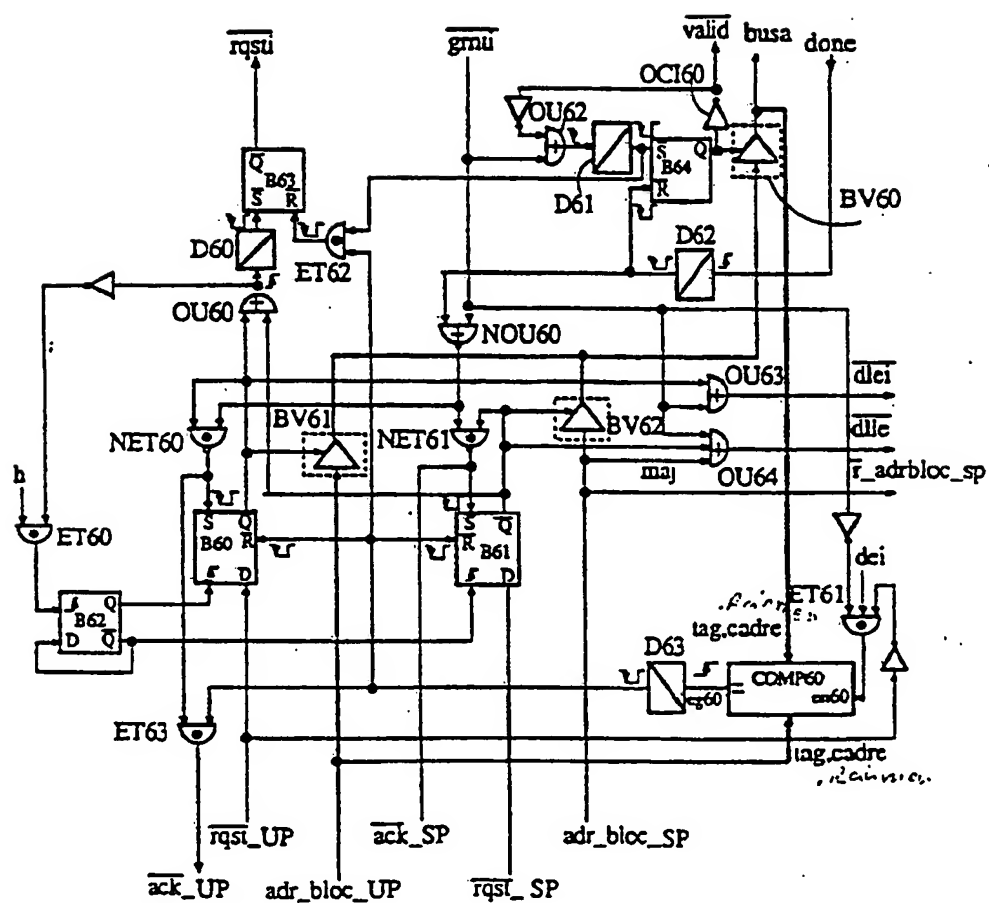
15/ 22

Fig. 15



16/22

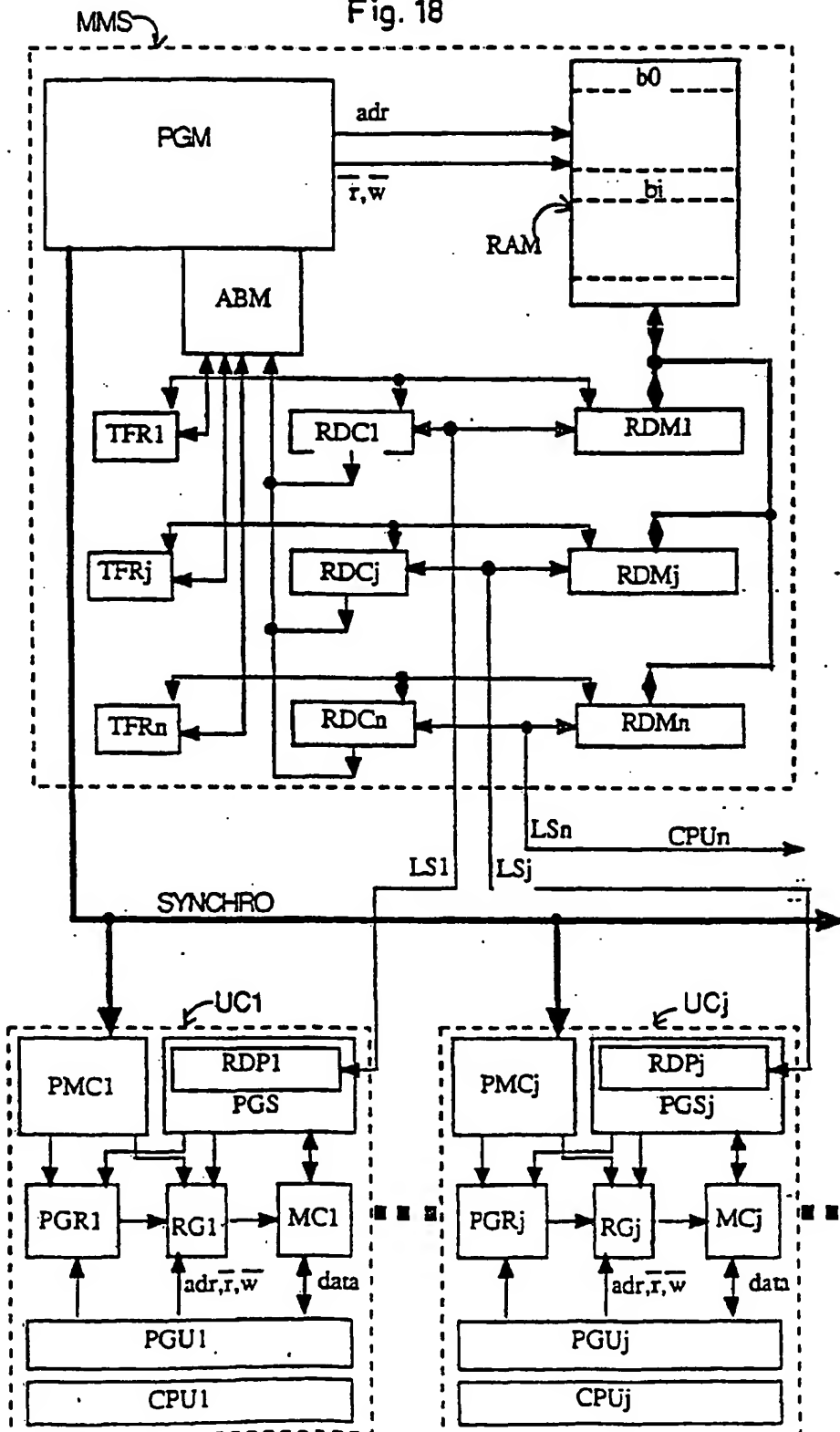
Fig. 16



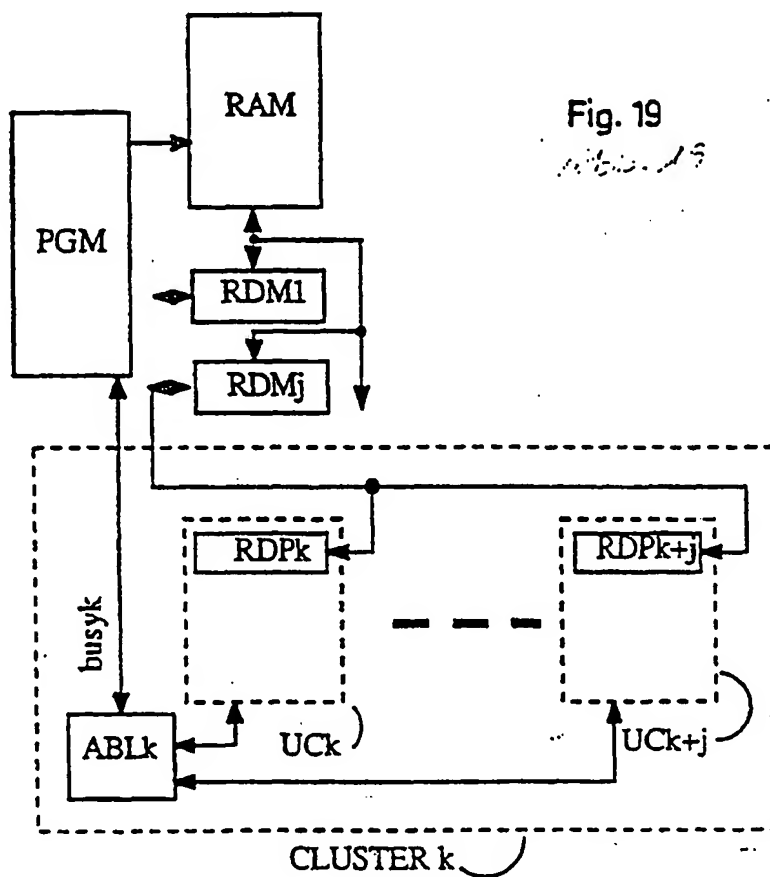


18 / 22

Fig. 18



19 / 22



20/22

Abb. 20 a

FIG. 20 a

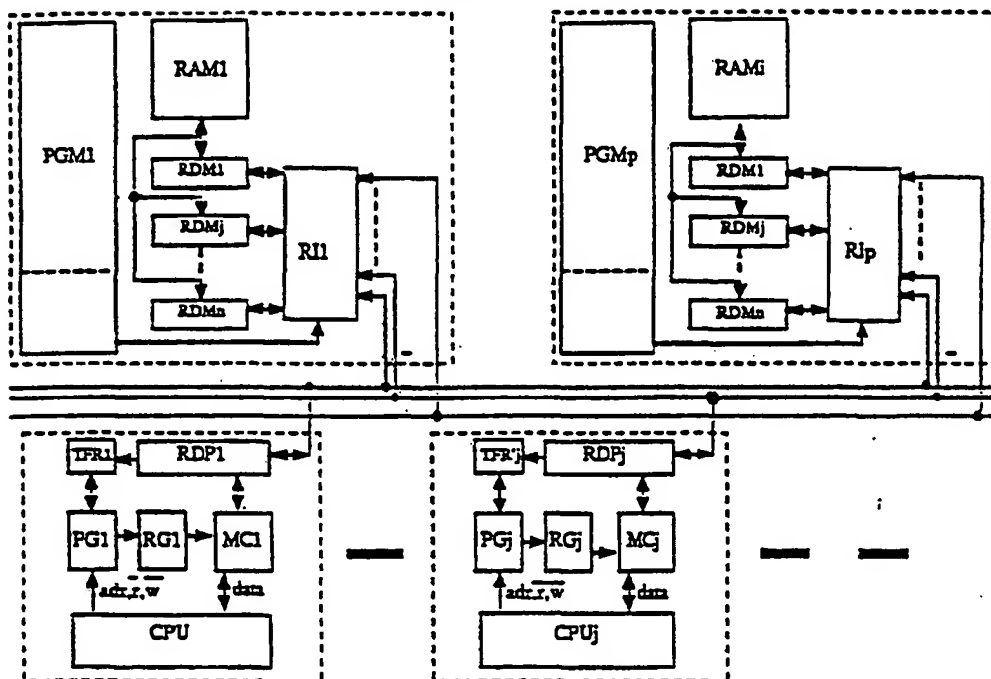
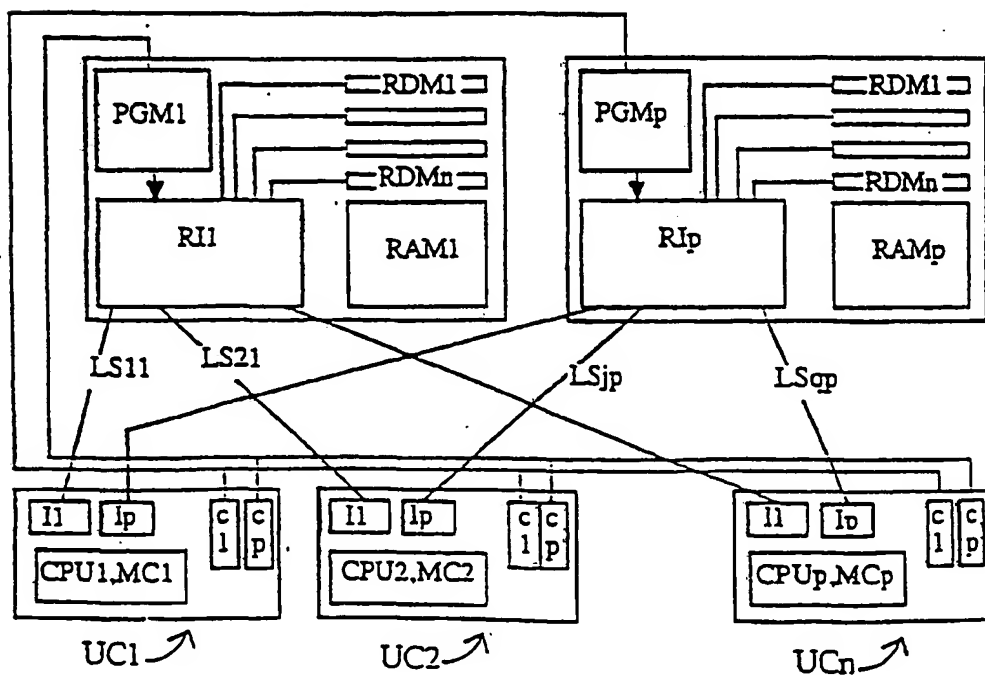


FIG. 20 b



21/22

Fig. 21a

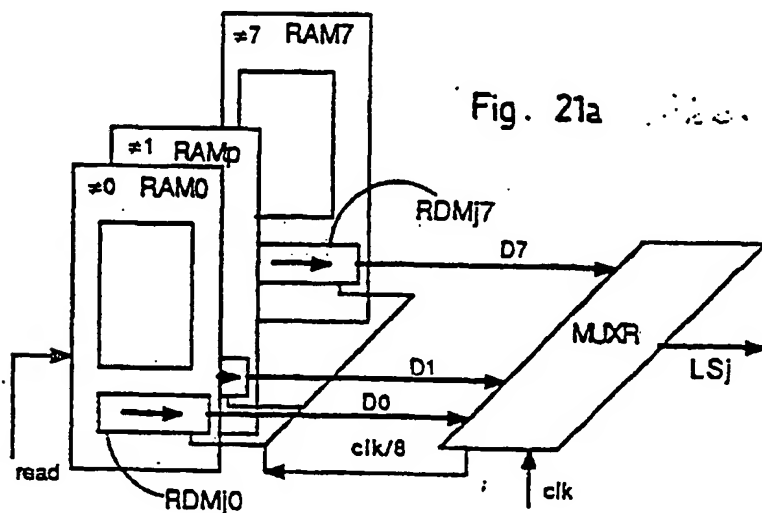
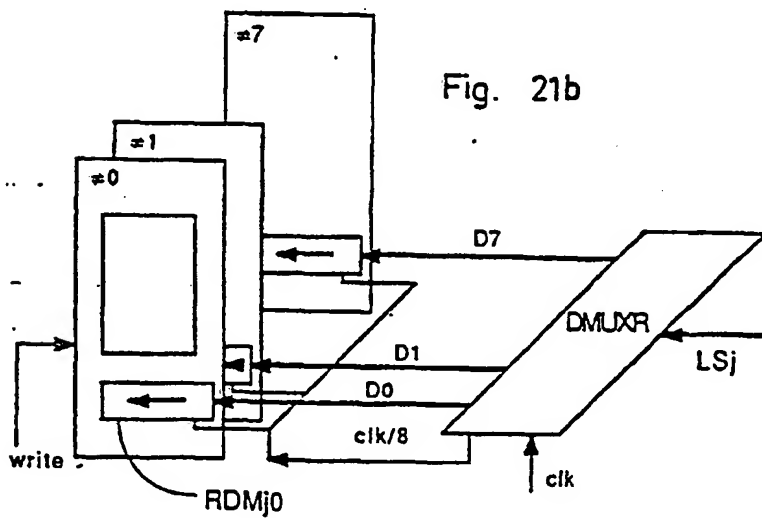


Fig. 21b



22/22

Fig.22 1/16.22

